Engineering with Computers (2024) 40:493–525
https://doi.org/10.1007/s00366-023-01802-1

ORIGINAL ARTICLE

An enhanced approach for inner and outer faces recognition of complex thin-shell parts

Pradiktio Putrayudanto[1] · Yi-Zhong Hwang[1] · Jiing-Yih Lai[1] · Pei-Pu Song[2] · Yao-Chen Tsai[2] · Chia-Hsiang Hsu[2]

Received: 10 January 2023 / Accepted: 23 February 2023 / Published online: 18 March 2023

Abstract
Volume decomposition is a technique to decompose a computer aided design (CAD) model into sweepable subvolumes, by which better types of mesh can be generated. A thin-shell part can be divided into a main body (thin shell) and protrusions that reside on the thin shell. When the thin shell and protrusions are separated, it would become easy to decompose each of them individually. Direct recognition of protrusions on a thin-shell part is error-prone owing to the complex structure of protrusions. The recognition of inner and outer faces on the thin shell can help the recognition of protrusions, as well as the volume decomposition of the thin shell. In this study, the complexity of the models considered includes the following: (1) various types of transition faces between inner and outer faces, (2) complex protrusion structures both on the inside and outside of the model, (3) fillets are included, and (4) complex holes lie across multiple faces. The proposed approach is divided into the following four steps: separation of inner and outer faces, transition faces recognition, inner faces recognition, and outer faces recognition. A detailed discussion of the procedures for each of the steps is provided. Also, 25 thin-shell models are employed to demonstrate the feasibility of the proposed method.

Keywords Inner and outer faces · Protrusion recognition · Feature recognition · Thin-shell part · Volume decomposition

1 Introduction

In mold flow analysis, a computer aided design (CAD) model must be converted into solid meshes for the analysis solver. Tetrahedral (Tet) mesh elements are commonly used because they are easy to generate and can be fitted for any complex geometry. However, the mesh element must be close to isotropic for achieving good mesh quality, which can result in excessive number of elements, especially for thin-shell components. On the contrary, hexahedral (Hex) or prismatic (Prism) meshes can be anisotropic while preserving good mesh quality, which indicates that the number of elements in the shell and thickness directions can be independent. However, to apply Hex or Prism elements, the decomposition of a CAD model is necessary. Although significant research on automatic Hex meshing of CAD models has been carried out, most of the methods are restricted to block structures [1–9]. As fully Hex meshing is difficult for complex components, research on semi-automatic Hex-dominant meshing has also be investigated [10–15]. However, automatic decomposition of CAD models is still a bottleneck owing to the complexity of CAD models in real applications.

Thin-shell plastic parts are used in many products and are frequently manufactured by injection molding. On both the inside and outside of this type of part, various features (such as protrusions and depressions) areattached to a main body (referred to as the thin shell). The thin shell and the features that reside on it can be complex in shape. The inner side is typically more complex in structure as it has many functional and structural features design. However, complex features may exist on the outside too. In addition, a CAD model usually involves fillets that are tiny or long and narrow to some extent. Fillet is one of the most troublesome issues in meshing as its dimension is usually much smaller than its neighboring faces. To improve the accuracy of mold flow analysis, users would rather perform mesh generation manually, as they are free to design and adjust meshes region by region. However, this requires considerable efforts in planning, model decomposition, and mesh generation.

✉ Jiing-Yih Lai
jylai@ncu.edu.tw

1 National Central University, Taoyuan, Taiwan

2 CoreTech System (Moldex3D) Co., Ltd., Hsinchu, Taiwan

For the decomposition of thin-shell CAD models, several approaches in the literature could be addressed. As a thin-shell part is mainly composed of a thin shell and various types of feature residing on it, it could be possible by applying feature recognition to identify the features directly. Feature recognition has been studied for many years. However, most of the methods are for the applications in CAD, CAM and CAPP [16–20]. Lu's method [1] based on the extraction of various types of edge loops can extract block shapes from a CAD model for Hex meshing. Wu's method [2] based on sweep direction extraction can divide a CAD model into several blocks that can be meshed by sweeping. However, they both claimed that their algorithms were valid only for limited cases. For automatic Hex meshing, the approaches of 3D cross field [3–6] and polycube [7–9] can convert smooth surface triangular meshes of an object into all Hex meshes. These methods are suitable primarily for block structures too. For semi-automatic Hex-dominant meshing, Robinson and Armstrong's approach [10–14] for thin-walled parts is much close to the issue addressed in this study. Their application is mainly for aerospace parts, e.g. engine cases, which involve the recognition and meshing of thin-shell and long-slender regions. However, the remaining complex regions are still processed manually.

The concept proposed in this study for solving the decomposition problem for thin-shell CAD models is to separate the thin shell and protrusions individually. Protrusions on different CAD models could be complex and variable. Traditional feature recognition methods by recognizing protrusions directly are error-prone. Both edge loops extraction [1] and sweep directions extraction [2] that directly extract block shapes from a model could easily result in overflow for complex components. This is why these two methods can only be applied for limited cases. In this study, the faces belonging to the thin shell are recognized and classified first. They are then used as the constraints for recognizing protrusion faces group by group. As the faces on each group are less complex in structure, the recognition and classification of protrusions become easier. The contours that each group of protrusion faces lie on the thin shell can also be evaluated. Once the faces of the thin shell and protrusions are separately obtained, it would be possible to achieve automatic volume decomposition of complex thin-shell models.

Recognition and classification of the faces on the thin shell is a critical issue for the aforementioned method. A typical thin-shell part can be divided into inner and outer faces. Faces that can be seen from outside when two parts are assembled are called outer faces, while those that cannot be seen are called inner faces. Inner faces mainly include faces that belong to the inner thin shell (including transition, wall and bottom faces; see Sect. 3) and the remaining protrusion faces. Similarly, outer faces also include faces that belong to the outer thin shell (including flange, outer wall and bottom faces; see Sect. 3) and the remaining protrusion faces. The face pair method proposed by Sun et al. [13] for extracting thin shell faces on thin-shell components could be employed. However, their method was only implemented on an "engine case", but has not been applied to other models. In addition, excessive face pairs that do not lie on the thin shell could be extracted as the conditions used for finding a face pair are relatively simple. Unlike most literature that only uses limited examples for demonstration, we collected many thin-shell CAD models in the beginning of this study to find out all kinds of situations that may occur.

In [21–23], we have developed algorithms for the recognition and decomposition of thin-shell CAD models. In [21], we focused on the recognition of ribs as ribs occupy most of the protrusions on a model. However, erroneous recognition result may still occur for complex models. In [22], we focused on the recognition of faces on the thin shell and employed them for the recognition of protrusion faces. In [23], we developed a volume decomposition algorithm for the thin shell by considering the matching of the meshes at the transition of the thin shell and protrusions. However, the complexity of the thin-shell models that can be processed was limited as follows: (1) the transition between the inner and outer faces is flat, (2) protrusions exist on inner faces only, and (3) other complex conditions are not considered.

The purpose of this study was to present an enhanced method for the recognition and classification of faces on the thin shell for more general thin-shell CAD models. In addition to the above-mentioned conditions, this study emphasizes the processing of the following conditions: (1) complex transition between inner and outer faces, (2) protrusions exist both on the inside and outside, (3) fillets are included, and (4) complex holes across multiple faces are included. A detailed description of the above-mentioned conditions will be given in Sect. 2. The proposed face recognition and classification method can accurately evaluate the faces on the thin shell of different kinds of thin-shell parts. A detailed description of the procedures in each step of the proposed method is provided. Twenty-five CAD models and the recognition results are also presented to demonstrate the feasibility of the proposed face type recognition and classification method.

2 Literature review

Feature recognition on CAD, CAM, and CAPP are mainly divided into two approaches, graph-based and volumetric-based. For the graph-based approach, Joshi et al. [24] proposed an attributed adjacency graph (AAG) built on a B-rep model and employed heuristics for recognizing polyhedral features for machining. Corney et al. [25] constructed a face-edge graph (FEG) for 2 1/2D parts and proposed a procedure for recognizing holes and pockets. Bruzzone et al. [26] addressed the problem of extracting adjacency information from a description of a solid object in terms of the face-to-face composition (FFC) model. The problem related to the development of adjacency-finding algorithms was discussed. Lim et al. [18] presented an algorithm for partitioning protrusion and depression features (DP-features) on models with free-form surfaces. A vertex-edge-graph (VEG) of all of the candidate edges on the model was generated. For the volumetric-based approach, Waco et al. [27] proposed a method based on alternating sum of volumes with partitioning (ASVP) for extracting block volumes form a solid model. Sakurai et al. [28] presented a method based on maximal volume decomposition (MVD) for extracting the maximum volume on a model. Woo et al. [29] proposed a quantitative measure, called orthogonal bounding factor (OBF), for the detection of protrusion features on loops of concave edges. All the above-mentioned methods only show the ability of extracting interesting features or volumes from a model for the machining purpose. They cannot be used for the decomposition of the entire model.

Liu et al. [30] employed CLoop (convex and concave loops) proposed in Gadh et al. [31] to decompose a solid model into multiple block structures, each of which represents a simple block shape, and then apply a transfinite mapping method that maps a unit cube into a Hex region in 3D space to convert each block shape into Hex meshes. Lu et al. [1] extended CLoop into PLoop (pure convex or concave loop), SLoop (mixed convexity, closed link), and HLoop (mixed convexity, open link), and investigated the partition of the 3D model based on these types of loop. A separator is one or more loops which can bound a cutting face to separate the model. Heuristic rules were proposed for generating separators and the corresponding cutting face for each of them was described. The limitation of this method is that the loop-based block volume recognition is not robust, especially for complex parts that contain loops beyond the consideration of this study.

In mesh generation, one of the approaches is to idealize a solid model with mid surfaces to apply shell elements on some regions that can be described by mid surfaces. Depending on the range of a model that is represented by mid surfaces, the resultant model can be a pure mid-surface model or a mixed-dimensional model combining mid surfaces and the remaining complex regions. This technique can greatly reduce the number of elements and hence the degrees of freedom required for meshing. Sheen et al. [32] proposed a solid deflation method to shrink a solid model into a very thin solid, and then convert it into a mid-surface model. This technique is applicable only to limited types of models with planar and quadratic surfaces. Zhu et al. [33] proposed a mid-surface abstraction method for thin-walled models based on rib feature decomposition. It defined a hierarchical semantic structure to describe the connection relationships between sub-regions and the affiliation relationship of two connected sub-regions. The rib features on the thin-walled model were identified and organized to form a hierarchical semantic structure. A model decomposition algorithm was then employed to decompose rib features in accordance with the hierarchical semantic structure. The mid-surface patches for each sub-region were finally abstracted through an adaptive abstraction method. Only three models were demonstrated in this study. More complex models need to be tested for verifying the robustness of this method. Robinson et al. [10] proposed a decomposition process based on the medial axis transform (MAT) for idealizing the thin-sheet regions of thin-walled structures as mid-surfaces, which can then be meshed using shell elements. The remaining complex regions on the 3D model were then meshed with tetrahedral meshes. Makem et al. [11] extended Robinson's work to find the long slender regions on thin-walled structures. It employed shape metrics generated using local sizing measures to identify long slender regions within the thick body. A series of algorithms were then applied for partitioning the think region into a non-manifold assembly of long/slender and complex sub-regions, which were then meshed with structured anisotropic meshes and unstructured isotropic tetrahedral meshes, respectively.

Nolan et al. [12] focused on the creation of a dimensionally reduced model for the purpose of structural analysis in the preliminary design and optimization stage of a thin-walled product. The 3D thin-walled CAD model was divided into thin-sheet, long-slender and complex regions. The thin-sheet regions were first identified using the method described in Robinson et al. [10]. The remaining thick regions were then sub-divided into long-slender and complex regions using the method described in Makem et al. [11]. The decomposed model was a non-manifold assembly of thin-sheet, long-slender and complex volumes, which were then meshed using shell, beam and tetrahedral elements, respectively. Sun et al. [13] proposed a face-pair based method for identifying thin-sheet regions, which is computationally more efficient than the MAT method described in Robinson et al. [10]. Instead of extracting mid surfaces from the face pairs, this method focused on how to decide the boundaries of the target thin-sheet regions and how to create cutting faces to decompose the models without generating silver volumes in either thin-sheets or residual domain. Each set of face pair can form a sweepable region that can be meshed with hexahedral meshes. Sun et al. [14] later proposed an enhanced method for long-slender region identification. It also emphasized the decision of the boundaries of the target regions and the corresponding cutting faces to isolate the long-slender regions suitable for sweep meshing.

For automatic Hex meshing, sweeping is one of the most robust techniques to generate Hex meshes. White et al. [34]

proposed a CCSweep technique to automatically decompose multi-sweepable volumes into many-to-one sweepable volumes. It converted multiple-source-and-target-faces into a single-target-face problem, enabling the sub-volumes to beautomatically meshed using a many-source-to-one-target Hex sweeping approach. Cai et al. [35] addressed the one-to-one sweeping and indicated that the most difficult problem was to map an all-quad source surface mesh onto its target surface. They proposed a harmonic function for the morphing of the meshes on a source surface onto its target surface, and a cage-based method for locating nodes inside volumes. This algorithm can map surface meshes between two concave or multiply-connected surfaces, and can also deal with geometries with twisted and complicate boundaries. Wu et al. [36] proposed an automatic swept volume decomposition technique based on sweep directions extraction. It first extracted all potential local sweep directions from the model and generated relevant face sets for each of them. The reasonable cutting face set that can split the model into swept sub-volumes was then constructed. Finally, the relevant face sets were used to generate maximal single-axis swept subvolumes. However, based on the decomposed results provided, it seems that some subvolumes are still complex in shape and may not be meshed by sweeping.

Research on automatic Hex meshing for an object represented as smooth Tet surface meshes has also been studied extensively. Luo et al. [37] proposed a method to obtain a near optimal finite element mesh from a coarse Tet surface mesh of a CAD model. It can identify thin sections of the model through a set of discrete medial surface points computed from an Octree-based tracing algorithm and convert Tet elements into Prism elements in the thin directions. It can also identify geometric singular edges and generate geometrically graded meshes from the edges. The meshes can then be mapped onto the geometry to the required level. Liu et al. [38] presented a method based on skeleton-based polycube generation to construct feature-preserving T-meshes. From the input skeleton of a model, initial interior cubes and boundary cubes that contact with the outer surface were created. Each cubic region was then subdivided to obtain T-spline control mesh. During the subdivision, the mesh boundary was aligned to preserve surface features, which include open curves, closed curves and singularity features. The T-meshes were finally extracted as Bezier elements for isogeometric analysis. Hu et al. [7] proposed an automatic polycube construction algorithm using harmonic boundary-enhanced centroidal Voronoi tessellation (HBECVT) based surface segmentation. Given a smooth surface triangle mesh, the polycube construction was viewed as a mesh segmentation task. The HBECVT method introduces local neighboring information into the energy function, which can reduce non-monotone boundaries and is less sensitive to the noise. Based on the constructed polycube, uniform Hex meshes, T-spline control meshes and adaptive all-Hex dual meshes could be generated. Hu et al. [8] modified the HBECVT method by introducing eigenfunctions of the secondary Laplace operator for surface segmentation and a novel generalized harmonic boundary-enhanced CTV model for polycube construction. This modified method can reduce the computational cost and eliminate unsmoothed boundary and over-segmentations.

Yu et al. [9] presented a software package, HexGen and Hex2Spline, to integrate geometry design with isogeometric analysis in LS-DYNA. Given a CAD model, HexGen creates a Hex mesh by using a semi-automatic polycube-based mesh generation method. Hex2Spline can construct hierarchical splines by using the Hex mesh from HexGen. HexSpline can also transfer spline information to LS-DYNA and performs isogeometric analysis. Yu et al. [15] further presented a Hex-Dom software package that can create a Hex-dominant mesh in real applications. A semi-automated polycube-based mesh generation method was employed. The resulting mesh is Hex dominant, but is also composed of Tet and Prism meshes.

3 Basic concept and method overview

A thin-shell CAD model is basically composed of a thin shell that forms the basic shape of the part. The faces on a part can be divided into inner and outer faces, where the former are invisible when two parts are assembled and the latter are visible. The inner faces can be divided into the following four types (Fig. 1a): transition, wall, bottom, and protrusion faces. Faces that are adjacent to outer faces are called transition faces. Faces that form the bottom of the inner region are called bottom faces. Faces on the side wall are called wall faces. The remaining faces belong to protrusion faces. When multiple layers of wall face exist, the upper layer is called wall faces, while the lower layers are called step-wall faces, as shown in Fig. 1b. Similarly, the outer faces can be divided into the following four types (Fig. 1c): outer wall, outer bottom, protrusion and flange faces. The definition of outer wall, outer bottom and protrusion faces is similar to those on inner faces. Faces that are between outer wall and transition faces are called flange faces. Flange faces may not exist on a model.

Thin-shell CAD models can be analyzed in accordance with the following conditions. First, a transition face is basically a face that directly connects to inner and outer wall faces simultaneously. However, complex transition faces with different kinds of cross section may exist. Five types of transition face are commonly occurred, which are: (1) simple transition (Fig. 2a): the cross section is composed of a line or a line and arcs that connect to the line smoothly. The transition faces include a face or a face and its neighboring fillets. It is the most common type; (2) step (Fig. 2b):

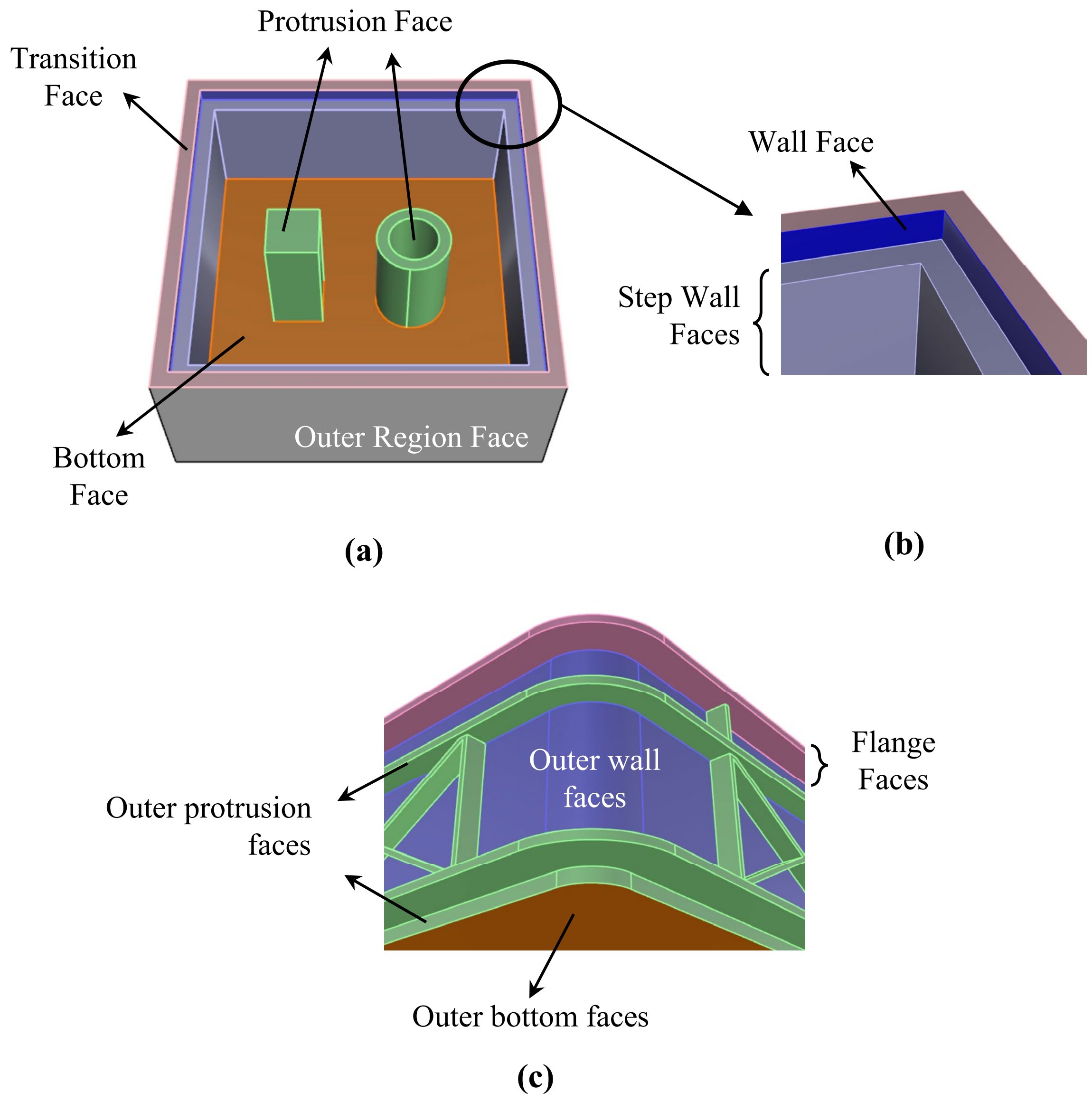


Fig. 1: Composition of faces on thin-shell parts, **a** inner faces, **b** multiple-layer inner wall, and **c** outer faces

the cross section is a step. The transition faces include two faces of different heights and a face that connects to them; (3) open step (Fig. 2c): it is like a "step", but with a sudden jump to become a "simple translation" in some regions; (4) extruded ridge (Fig. 2d): the cross section involves a convex ridge. The transition faces include the faces on the ridge and its neighboring faces; and (5) depressed ridge (Fig. 2e): the cross section involves a concave ridge. The transition faces include the faces on the ridge and its neighboring faces.

Second, the protrusion structure on the outside is more complex than that on the inside. Both outer wall and outer bottom faces can be divided into concave and non-concave types, as shown in Fig. 3. When most of the protrusions on the outside are distributed continuously and partition the wall faces into many regions, this kind of part is called a concave wall type (Fig. 3a). On the contrary, when most of the protrusions are distributed sparsely, it is called a non-concave wall type (Fig. 3b). In addition, the outer bottom face is normally lower than its neighboring wall faces, e.g. Fig. 3a, b). However, it may occur that the outer bottom face is sunken inside and becomes higher than its neighboring wall faces, e.g. Fig. 3c. The former is called a non-concave bottom type, while the latter is called a concave bottom type. Such a classification is helpful for the recognition of different face types.

Finally, fillets and holes are common features on CAD models. They are usually neglected or simplified for the simplification of the problem. However, in real

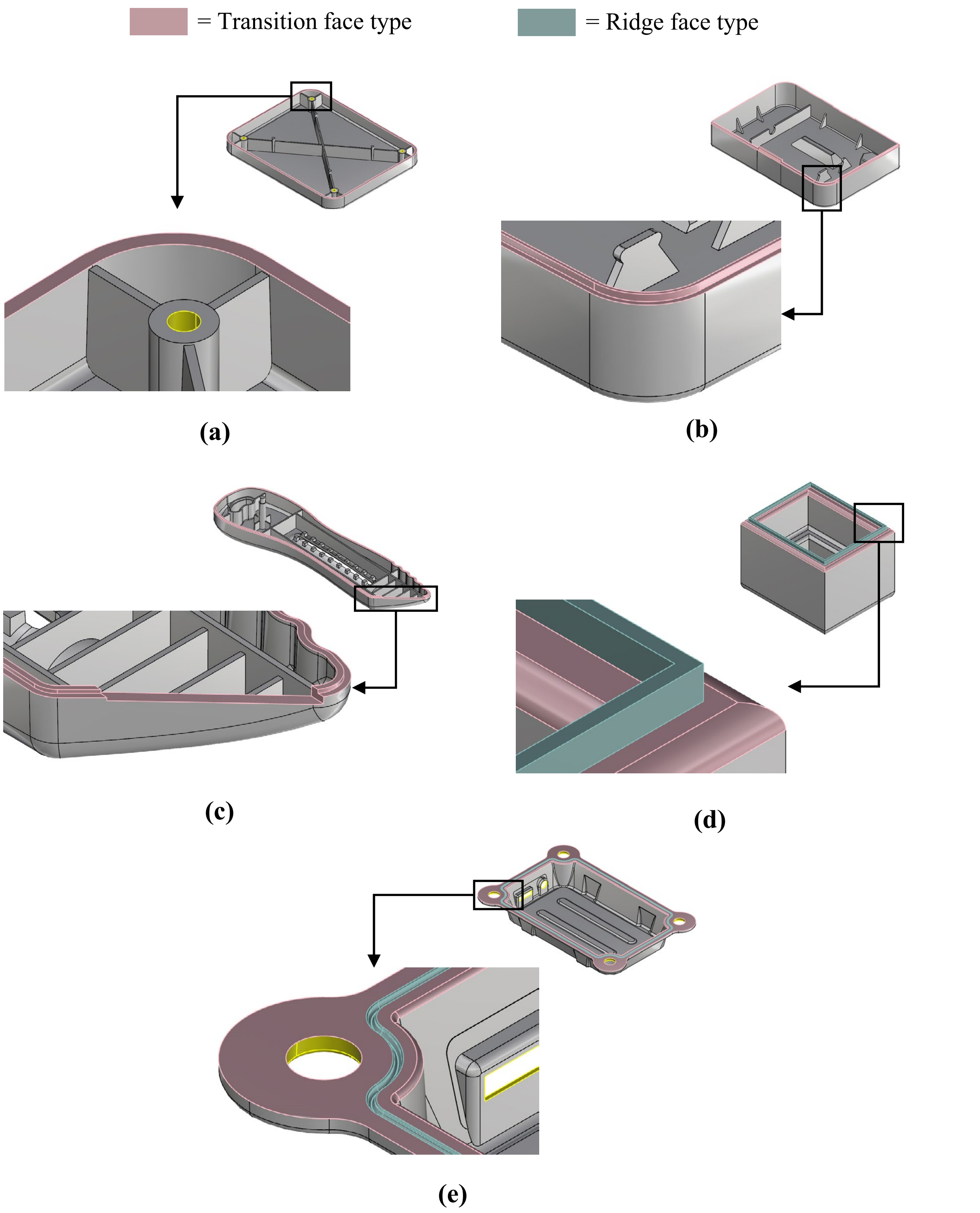


Fig. 2 Five types of transition faces, **a** simple transition, **b** step, **c** open step, **d** extruded ridge, and **e** depressed ridge

CAD models, these features always exist and should be addressed too. When classifying faces on a thin shell into different face types, the belonging of the fillets that connect to these faces should be determined also. In addition, most holes are located on a simple face only. However, holes that lie across multiple faces may also exist and complicate the face type recognition problem. Therefore, the

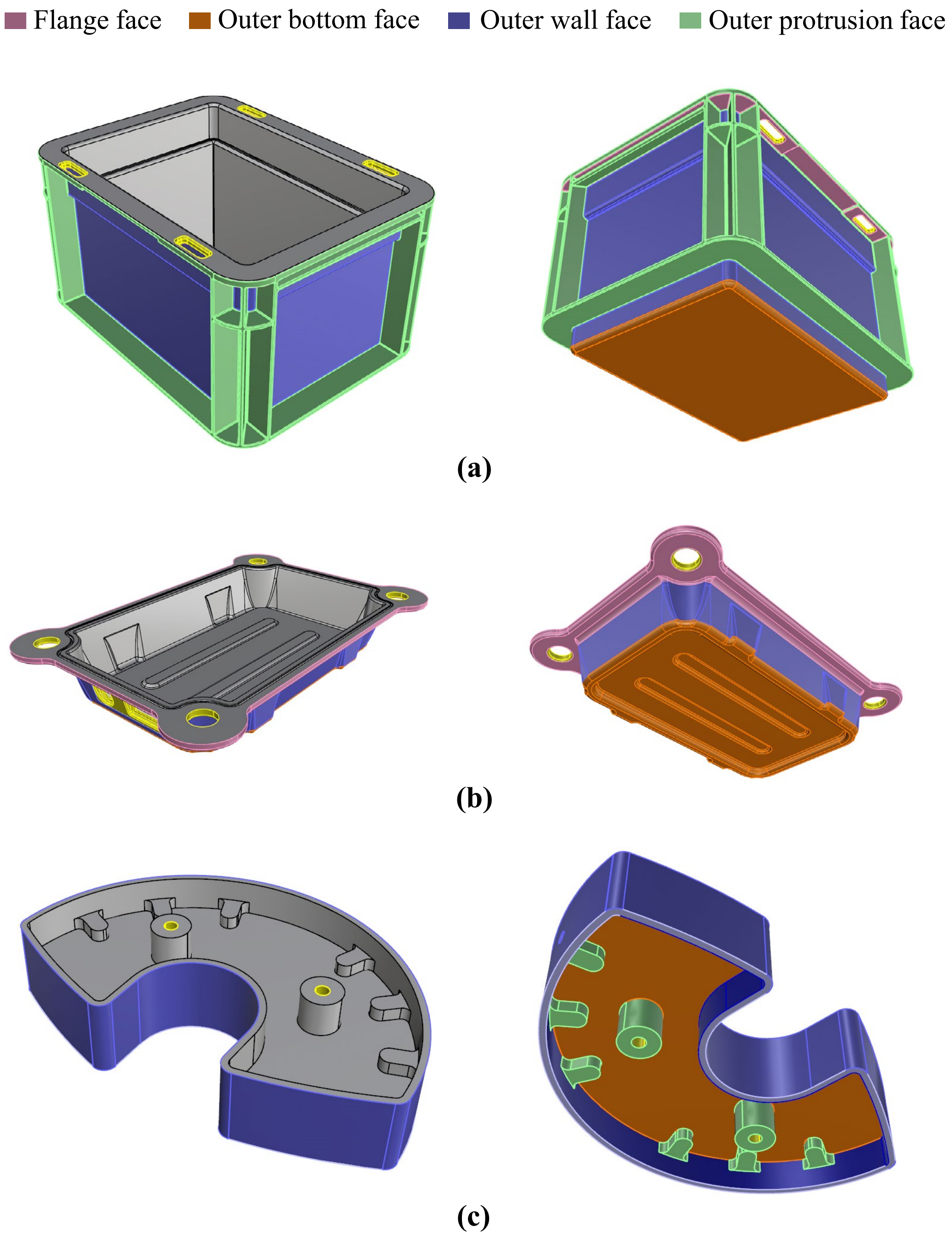


Fig. 3 Three types of wall and bottom faces on the outside, **a** concave wall, **b** non-concave wall, and **c** concave bottom

issues related to fillets and holes should be addressed and solved also.

In this study, an enhanced inner and outer faces recognition algorithm is developed to cover all kinds of thin-shell CAD models, as mentioned above. Figure 4 shows the overall flowchart of the proposed face type recognition algorithm. It can mainly be divided into the following two steps: preliminary functions and face type recognition. The input is a boundary representation (B-rep) model of the thin-shell part, and the output is the composition of faces both on the inside and outside. The main improvement of this study, compared with that in [22], is that the enhanced method can cover five types of transition face, fillets, complex holes, and concave and non-concave types both on outer wall and bottom faces. This study primarily focuses on the development of the face type recognition. However, preliminary functions are briefly described below for completeness of the method.

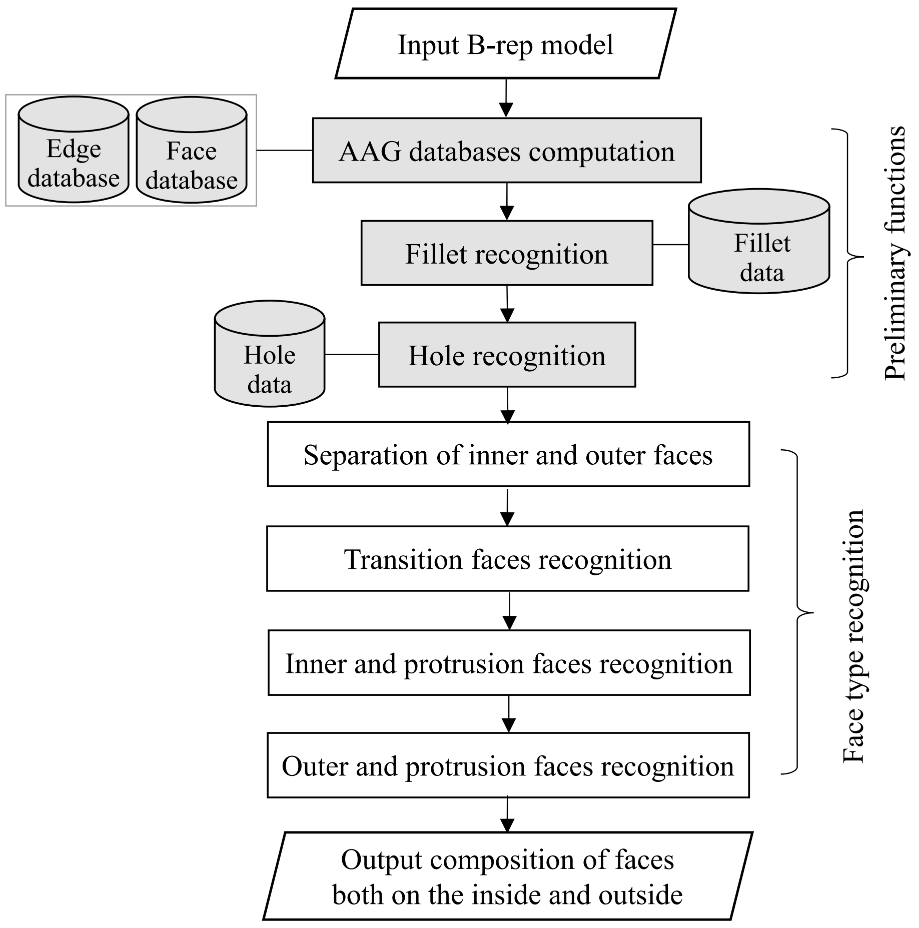

Input B-rep model
Edge database
Face database
AAG databases computation
Fillet recognition
Fillet data
Hole data
Hole recognition
Preliminary functions
Separation of inner and outer faces
Transition faces recognition
Inner and protrusion faces recognition
Outer and protrusion faces recognition
Face type recognition
Output composition of faces both on the inside and outside

Fig. 4 Flowchart of the proposed face type recognition method for thin-shell CAD models

In preliminary functions, the edge and face databases are computed first. Each database records the geometric and topological data of the associated entity. Fillet recognition is implemented next, which outputs the compositions of edge blended faces (EBF) and vertex blended faces (VBF). Fillet data enable the evaluation of faces across any EBF or VBF. Next, hole recognition is implemented. Hole recognition allows all blind and through holes on the B-rep model to be recognized. Coaxial holes can also be detected by this algorithm. More importantly, holes that lie across multiple faces are also recognized. It relies on the recognition of virtual and multi-virtual loops first. A virtual loop is a closed contour formed by multiple faces that connect at least G^1 continuous, whereas a multi-virtual loop is also a closed contour by multiple faces, but with some junctions G^0 continuous.

In face type recognition (Figs. 140, 150, 160), inner and outer faces are separated first. It can normally yield a closed loop of parting line representing the transition of inner and outer faces. Transition faces are next determined. For "simple transition" (Fig. 2a), the transition faces are directly adjacent to the parting line. However, to cover the other four types of transition face (Fig. 2b–e), an algorithm is employed to search the faces that can represent the cross section of transition faces and determine the transition type. The face types on inner and outer faces are next recognized, respectively. For inner faces, a smooth face is often partitioned into many pieces owing to the complex structure of protrusions on the inside. Fillets may also appear at the junction of two or several faces and complicate the determination of the face type. Specific rules are provided to recognize wall and bottom faces in sequence. With transition, wall and bottom faces recognized, protrusion faces can be recognized and divided into groups based on the adjacency relationship. For outer faces, wall, bottom and flange faces are recognized in sequence. Both outer wall and bottom faces are divided into two types: concave and non-concave types. The distinction of concave and non-concave bottom types is used for the recognition of wall, bottom and flange faces; while the distinction of concave and non-concave wall types is used for the recognition of protrusion faces.

4 Proposed face type recognition method

4.1 Separation of inner and outer faces

Consider that the part model is aligned so that the mold opening direction is along $+Z$, and the two axes of its horizontal plane are parallel to the X and Y directions, respectively. The other five faces on the bounding box of the part model are perpendicular to $+X$, $-X$, $+Y$, $-Y$ and $-Z$, respectively. Based on the definition on inner and outer faces previously, the outer faces are visible from the outside, whereas the inner faces are invisible. The visibility of a face could be detected by projecting lines from this face onto the boundary planes and checking the intersection of these lines with the other faces [22]. However, this concept could be valid for simple shape, but may not be applicable for complex geometry, especially when protrusion features exist on outer faces. Figure 5 shows the flowchart of inner and outer faces separation. The faces are initially divided into three types based on the visibility from + Z. Some of the face types are modified in accordance with the adjacency relationship. One or several loops of transition edges that represent the boundary of inner and outer faces can then be obtained. The inner and outer faces are finally separated based on the main loop of transition edges. The detailed algorithm is described below.

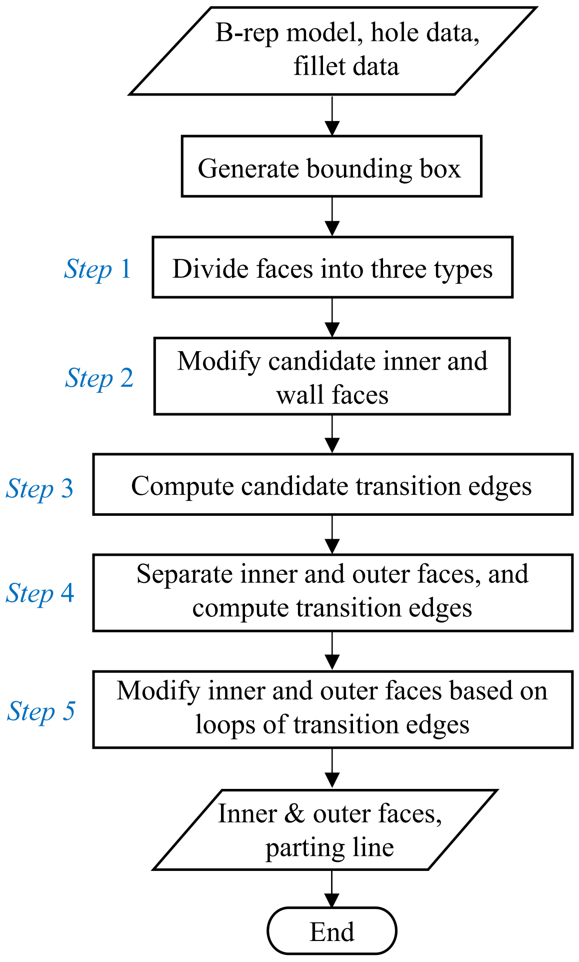

B-rep model, hole data, fillet data
Generate bounding box
Step 1
Divide faces into three types
Step 2
Modify candidate inner and wall faces
Step 3
Compute candidate transition edges
Step 4
Separate inner and outer faces, and compute transition edges
Step 5
Modify inner and outer faces based on loops of transition edges
Inner & outer faces, parting line
End

Fig. 5 Flowchart of inner and outer faces separation

4.1.1 Step 1: Divide faces into three types

The faces are initially divided into candidate inner, wall and outer faces by checking the intersection with the boundary box. Consider one face f_i on the model. If f_i meets the following conditions, then it is regarded as a candidate outer face: (1) $\theta_f > 170°$, where θ_f is the angle between the surface normal $\boldsymbol{n}_f$ and + Z; (2) L_f does not intersect any face on the model, where L_f is a line from the centroid of f_i towards $\boldsymbol{n}_f$; and (3) f_i is not a hole face of through holes. When f_i is neither a candidate outer face nor a hole face of through holes, it is assigned to one of the following two regions: (1) Region I: $0° \le \theta_f \le 90° + \varepsilon$, and (2) Region II: $90° + \varepsilon < \theta_f \le 180°$, where ε is a draft angle, 3° in this study.

When f_i is in Region I, the candidate inner and wall faces are determined based on a projecting-line intersection check and face adjacency relationship. Define two parameters $\mathrm{Max}\left(\boldsymbol{n}_b\right)$ and C_2, where $\mathrm{Max}\left(\boldsymbol{n}_b\right)$ denotes one of the six directions $\pm X$, $\pm Y$, $\pm Z$ that is closest to the direction of $\boldsymbol{n}_f$, and C_2 is the status of intersection. Project a line from the centroid of f_i towards the direction $\mathrm{Max}\left(\boldsymbol{n}_b\right)$ and check if it intersects any face on the model. If an intersection occurs, then set C_2 as true, otherwise set C_2 as false. The face type of f_i is determined based on face adjacency relationship, as follows:

(1) If f_i is not adjacent to any candidate outer face, then regard f_i as a candidate inner face.
(2) If f_i is adjacent to a candidate outer face, $\mathrm{Max}\left(\boldsymbol{n}_b\right) \neq +\mathrm{Z}$, and $C_2 = \text{false}$, then regard f_i as a candidate wall face
(3) Otherwise, regard f_i as a candidate inner face. Also, mark f_i.

The reason to apply such complex rules is because some special cases cannot be judged by the angle θ_f only. The faces marked in Condition (3) will later be used to help determining the parting line. The left plot in Fig. 6a shows two faces that satisfy Conditions (1) and (2), respectively.

When f_i is in Region II, the candidate inner and wall faces are determined by projecting a line from the target face and check the number of intersections. Different face type has different number of intersections. Define a parameter C_5, where C_5 denotes the status of the intersection. Project a line from the centroid of f_i towards $\pm X$, $\pm Y$ and $-Z$, respectively, and check if it intersects any face on the model. If an intersection occurs, then C_5 is increased by 1 (The values of C_5: 0–5). The face type of f_i is determined as follows:

(1) If $C_5 = 5$, then regard f_i as a candidate inner face.
(2) Otherwise, regard f_i as a candidate wall face.

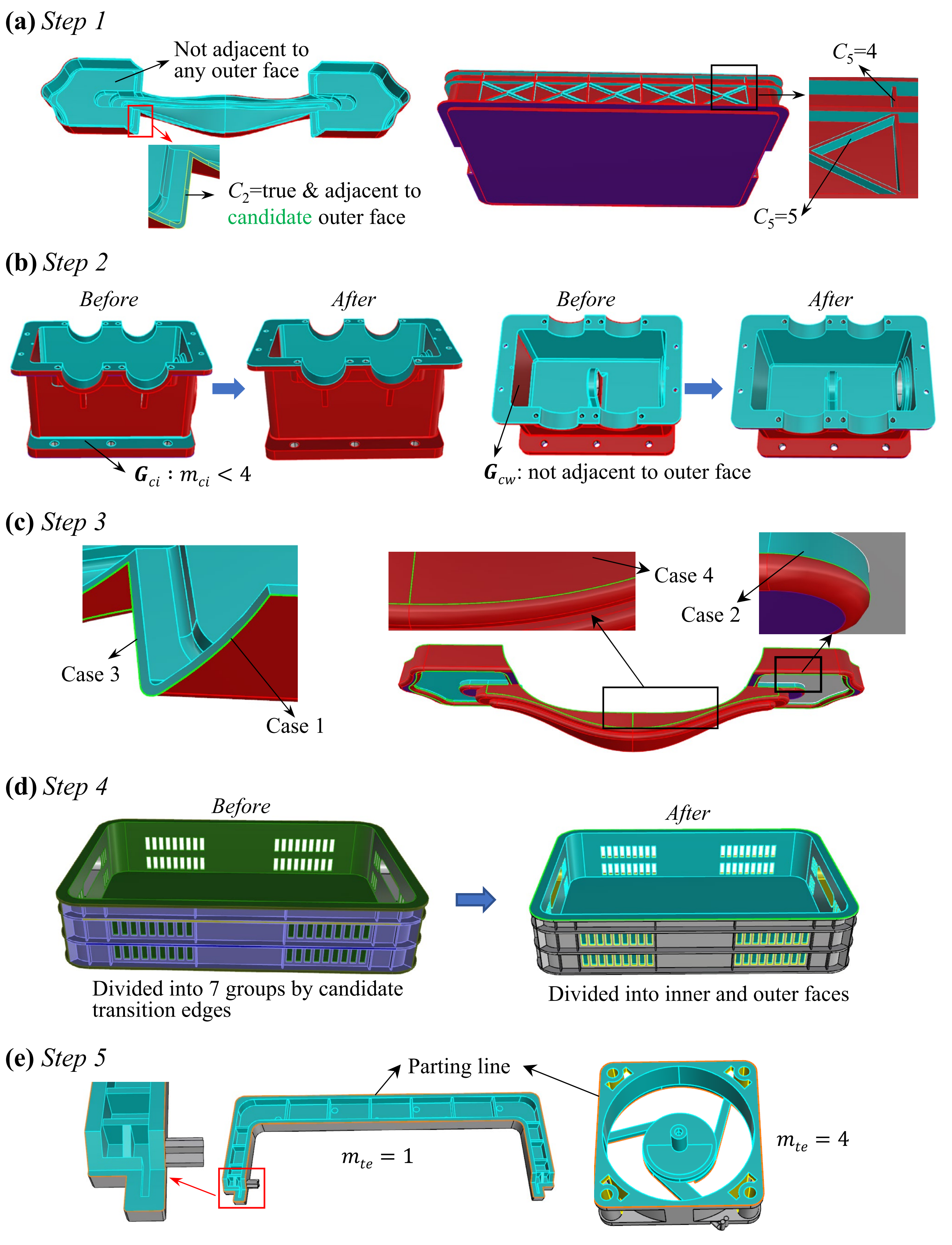

(a) Step 1
Not adjacent to any outer face
C_2=true & adjacent to candidate outer face
C_5=4
C_5=5
(b) Step 2
Before
After
Before
After
G_{ci} : m_{ci} < 4
G_{cw}: not adjacent to outer face
(c) Step 3
Case 3
Case 1
Case 4
Case 2
(d) Step 4
Before
After
Divided into 7 groups by candidate transition edges
Divided into inner and outer faces
(e) Step 5
Parting line
m_{te} = 1
m_{te} = 4

Fig. 6 Immediate results for inner and outer faces separation, **a** Step 1, **b** Step 2, **c** Step 3, **d** Step 4, and **e** Step5

The right plot in Fig. 6a shows two faces that are assigned as candidate wall ($C_5 = 4$) and inner ($C_5 = 5$) faces, respectively.

4.1.2 Step 2: Modify candidate inner and wall faces

Some groups of candidate inner and wall faces may wrongly be recognized in previous step. They are detected and modified in accordance with face adjacency relationship, as follows:

1. Group candidate inner faces $\boldsymbol{G}_{ci}$: all candidate inner faces that are adjacent to each other are regarded as a group $\boldsymbol{G}_{ci}$.
2. Check and modify candidate inner faces: Let f_i be one face in $\boldsymbol{G}_{ci}$. Project a line from the centroid of $\boldsymbol{G}_{ci}$ towards $\boldsymbol{n}_f$ and check if it intersects any face in $\boldsymbol{G}_{ci}$. Let m_{ci} be the intersection count for faces in $\boldsymbol{G}_{ci}$. When an intersection occurs, m_{ci} is increased by 1. If any group $\boldsymbol{G}_{ci}$ with $m_{ci} < 4$, then all faces in $\boldsymbol{G}_{ci}$ are changed to candidate wall faces.
3. Group candidate wall faces $\boldsymbol{G}_{cw}$: all candidate wall faces that are adjacent to each other are regarded as a group $\boldsymbol{G}_{cw}$.
4. Check and modify candidate wall faces: if all faces in $\boldsymbol{G}_{cw}$ are not adjacent to any candidate outer face, then all faces in $\boldsymbol{G}_{cw}$ are changed to candidate inner faces.

Figure 6b shows two groups of faces that meet the conditions in Procedures (2) and (4), respectively.

4.1.3 Step 3: Compute candidate transition edges

Transition edges represent the common boundary of inner and outer faces. The edges on all candidate inner faces are checked one by one to determine candidate transition edges. Let f_i be a candidate inner face and e_{ci} be an edge on f_i. The conditions for determining candidate transition edges are as follows:

(1) If e_{ci} is adjacent to a candidate outer face and convex, and f_i is marked, then e_{ci} is regarded as a candidate transition edge.
(2) If e_{ci} is adjacent to a candidate wall face and convex, then e_{ci} is regarded as a candidate transition edge.
(3) If e_{ci} is adjacent to a candidate wall face and one of its neighboring faces is a fillet, then e_{ci} is regarded as a candidate transition edge.
(4) If e_{ci} is adjacent to a candidate wall face and concave, then the following algorithm is implemented: search the neighboring faces of the candidate wall face in all directions. If the boundary edge is convex or the face is not a candidate wall face, then stop search in that direction. It finally yields a set of neighboring candidate wall faces that are convex in all boundary edges. The boundary edges are regarded as candidate transition edges.

Figure 6c shows four edges that represent the above-mentioned four conditions, respectively.

4.1.4 Step 4: Separate inner and outer faces, and compute transition faces

Separate all faces into two types: inner and outer faces, except through hole faces. The common boundaries of inner and outer faces are regarded as transition edges. Transition edges will form one or several loops. For a group of inner faces that are adjacent to each other, it should have at least one face that is facing up and has a projecting line that does not intersect any face on the model. The algorithm to separate inner and outer faces and evaluate transition edges is as follows:

(1) Divide faces into groups by candidate transition edges and edges of through holes: Faces that are adjacent to each other are regarded as a group, with candidate transition edges and edges of through holes being the boundary. It can yield multiple groups of face $\boldsymbol{G}_f$.
(2) Divide faces (except transition hole faces) into inner and outer faces: All faces in each group $\boldsymbol{G}_f$ are checked. Consider that f_i is a face in $\boldsymbol{G}_f$. Generate a line L_i from the centroid of f_i towards + Z. If the following two conditions are satisfied, then all faces in $\boldsymbol{G}_f$ are regarded as inner faces:

    (a) At least a face f_i that is nearly horizontal and facing up, i.e. $\theta_f \leq \varepsilon$.
    (b) At least a line L_i that does not intersect any face on the model.

Otherwise, all faces in $\boldsymbol{G}_f$ are regarded as outer faces.

(C) Compute transition faces: When an edge is the common boundary of an inner and outer faces, it is regarded as a transition edge.

Figure 6d shows an example divided into 7 groups by candidate transition edges. Only one group of faces are assigned as inner faces in this step.

4.1.5 Step 5: Modify inner and outer faces based on loops of transition edges

One or several loops of transition edges can be formed. By checking the number of loops and geometric and face adjacency conditions on each loop of faces, the face type can be determined. The procedures are as follows:

(1) Evaluate loops of transition edges: each set of adjacent transition edges that form a loop are recorded. It can yield one or several loops of transition edges $\boldsymbol{G}_{te}$. Denote m_{te} as the number of loops.
(2) Modify inner and outer faces based on m_{te}:

   (a) $m_{te} = 0$: ideally, at least one loop of transition edges can be found. If $m_{te} = 0$, it indicates that a model with ambiguous transition edges between inner and outer faces exists. The step in Sect. 4.1.3 must be implemented again, with Condition (1) modified as: (1') The edge exists between a candidate outer and inner face and is convex.
   (b) $m_{te} = 1$: only one loop of transition edges is found. It is the typical case. No modification of the inner or outer face is needed. The corresponding loop of transition edges is called the parting line herein.
   (c) $m_{te} > 1$: at least two loops of transition edges are found. The one with the longest length is the parting line, while the remaining loops must be checked. Consider that $\boldsymbol{G}_{te}$ is one of the remaining loops. Find all outer faces that are adjacent to $\boldsymbol{G}_{te}$. All such outer faces are expanded outside to form a face group $\boldsymbol{FG}_{te}$ until it reaches another loop of transition edges or through holes. Whether $\boldsymbol{G}_{te}$ should be preserved or not is determined as follows:

      (i) If FGtr contains at least a candidate outer face, then $\boldsymbol{G}_{te}$ is preserved.
      (ii) If $\boldsymbol{FG}_{te}$ does not contain any candidate outer face, then $\boldsymbol{G}_{te}$ is deleted.

If $\boldsymbol{G}_{te}$ is preserved, then it indicates that a large through hole (or pocket) exists on the model. On the contrary, if $\boldsymbol{G}_{te}$ is deleted, all faces inside this loop should be changed to inner faces.

Figure 6e shows two examples with $m_{te} = 1$ and 4, respectively. The parting lines and final inner faces are also displayed.

4.2 Transition faces recognition

As Fig. 2 depicts, transition faces are divided into the following five types: simple transition, step, open step, depressed ridge and extruded ridge. The "step" type may involve some faces from outer faces, while the faces on the other four types are all from inner faces. Therefore, the algorithm described below is used for detecting four types of transition faces only. The "step" type will be detected later in outer faces recognition. Figure 7 shows the flowchart of recognizing four types of transition faces, where the open step, extruded ridge and depressed ridge are detected in sequence. When none of the above three types is satisfied, it is regarded as a simple transition. In this algorithm, inner faces that are adjacent to the parting line are put into a group $\boldsymbol{G}_{dtr}$ first. The "open step" type has a sudden jump on neighboring transition faces. Therefore, it is detected by checking the edge concavity between neighboring transition faces in $\boldsymbol{G}_{dtr}$. For the "extrude ridge" and "depressed ridge" types, the faces in $\boldsymbol{G}_{dtr}$ are grown along the thickness direction to find a set of neighboring faces that can describe the cross-sectional shape of the transition. The detection of each transition type should be implemented individually as each has its own procedures of determining transition faces.

4.2.1 Open step type

Consider that f_z is the open bounding plane perpendicular to $+Z$. For each f_i in $\boldsymbol{G}_{dtr}$, if it is parallel to f_z, then the distance between f_i and f_z is computed. Denote the minimum distance among them as $d_{\min}$. All other inner faces that are parallel to f_z are then checked to find faces with a distance smaller than $d_{\min}$. All such faces are put into a group $\boldsymbol{G}_{ut}$, called upper transition faces. Check all faces f_n that are adjacent to the faces in $\boldsymbol{G}_{ut}$ and divide them into two groups. Generate a line f_n from the centroid of f_n towards its surface normal $\boldsymbol{n}_f$. If L_n intersects with a bounding plane, then put f_n into a group $\boldsymbol{G}_{\text{out}}$. Otherwise, if L_n intersects with an inner face, then put f_n into another group $\boldsymbol{G}_{\text{in}}$. The conditions for determining the "open step" type is as follows:

(a) If $\boldsymbol{G}_{ut}$ is empty, then the transition is not an "open step" type.
(b) If $\boldsymbol{G}_{ut}$ is not empty, then

– If all faces in $\boldsymbol{G}_{\text{out}}$ are adjacent to $\boldsymbol{G}_{dtr}$ and all faces in $\boldsymbol{G}_{in}$ are not adjacent to $\boldsymbol{G}_{dtr}$, then the transition is an "open step" type.
– Otherwise, the transition is not an "open step" type.

When the transition is an "open step" type, the faces in $\boldsymbol{G}_{dtr}$, $\boldsymbol{G}_{\text{out}}$ and $\boldsymbol{G}_{ut}$ are regarded as transition faces. Figure 8a shows the definition of $\boldsymbol{G}_{dtr}$, $\boldsymbol{G}_{\text{out}}$, $\boldsymbol{G}_{ut}$ and $\boldsymbol{G}_{in}$, and transition faces for the "open step" type.

4.2.2 Extruded ridge type

For recognizing extruded and depressed ridge types, a threshold d_w for the maximum width of the ridge allowed must be assigned. An extruded ridge can be detected by searching neighboring faces that are connected convexly.

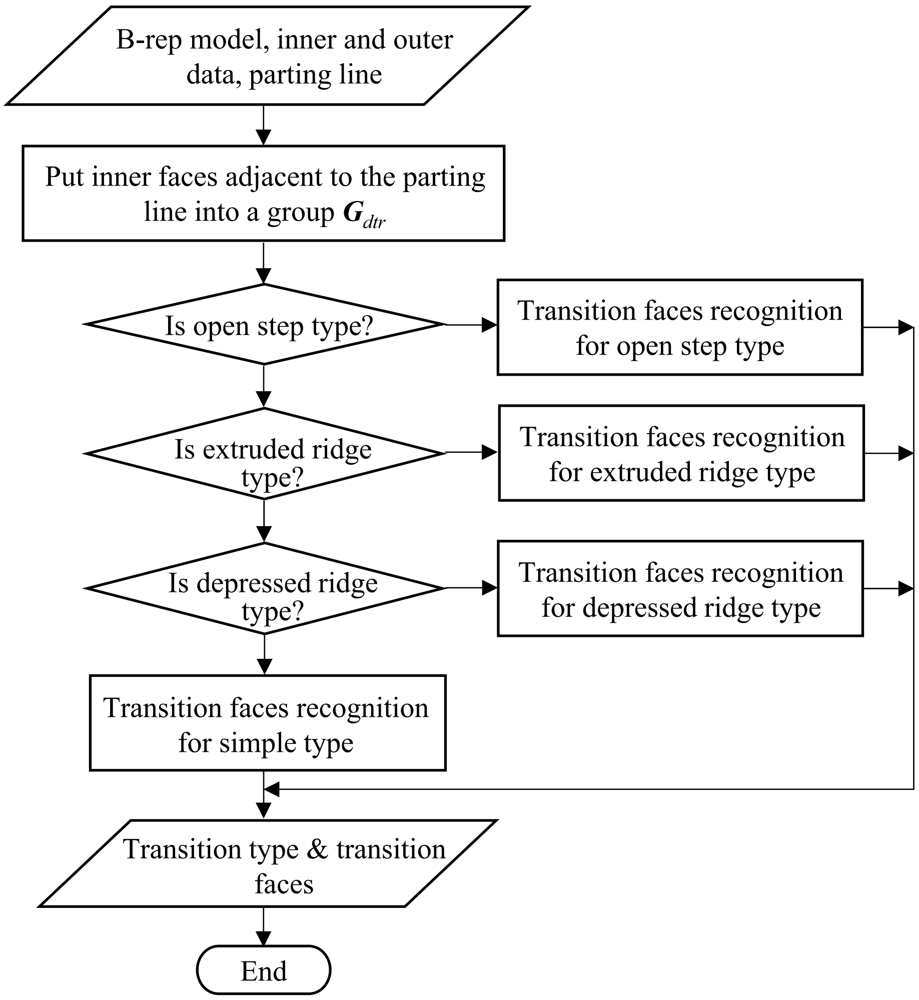

B-rep model, inner and outer data, parting line
Put inner faces adjacent to the parting line into a group $\boldsymbol{G}_{dtr}$
Is open step type?
Transition faces recognition for open step type
Is extruded ridge type?
Transition faces recognition for extruded ridge type
Is depressed ridge type?
Transition faces recognition for depressed ridge type
Transition faces recognition for simple type
Transition type & transition faces
End

Fig. 7 Flowchart of transition faces recognition

Put all faces in $\boldsymbol{G}_{dtr}$ into a face set $\boldsymbol{F}_{rs}$. For each of the faces in $\boldsymbol{F}_{rs}$, its neighboring faces are searched recursively along the thickness direction on both sides. Consider that f_i is a face in $\boldsymbol{F}_{rs}$ and f_n is one of the neighboring faces. If f_n meets the following conditions, then f_n is put into $\boldsymbol{F}_{rs}$: (1) f_i and f_n are connected convexly, and (2) $0° < \theta_f \le 90°$, where θ_f is the angle between the surface normal of f_n and $+Z$. Otherwise, the search stops on f_n. The search is continued for all faces in $\boldsymbol{F}_{rs}$ until all neighboring faces have been checked. The conditions for determining the "extruded ridge" type are as follows:

(a) Check every pair of vertical and parallel faces in $\boldsymbol{F}_{rs}$ and compute its distance, namely f_{rs} and f_{re} in Fig. 8b. For example, if there are four side walls on the thin shell, it will yield four pairs of vertical faces, and hence four distances. If all distances are smaller than d_w, then the transition is an "extruded ridge" type.
(b) Otherwise, the transition is not an "extruded ridge" type.

When the transition is an "extruded ridge" type, the transition faces are divided into three regions: (1) ridge faces: all faces in $\boldsymbol{F}_{rs}$. The fillets that are adjacent to the faces in $\boldsymbol{F}_{rs}$ are also included as ridge faces; (2) the horizontal face outside and its neighboring fillet; and (3) the horizontal face inside and its neighboring fillet. Figure 8b shows the definition of $\boldsymbol{G}_{dtr}$, $\boldsymbol{F}_{rs}$, f_{rs} and f_{re}, and transition faces for the "extruded ridge" type.

4.2.3 Depressed ridge type

Like the case of "extruded ridge", the detection of a "depressed ridge" is also started by searching neighboring faces convexly. However, it can only reach half of the cross-sectional shape as concave edges exist at the bottom of the depressed ridge. Start from a face f_i in $\boldsymbol{G}_{dtr}$, put all neighboring faces that are connected convexly into a face set $\boldsymbol{F}_{rs}$. In $\boldsymbol{F}_{rs}$, find a face f_{rs} with a face angle $90° \pm \varepsilon$, where ε is the draft angle. Generate a line from the centroid of f_{rs} towards

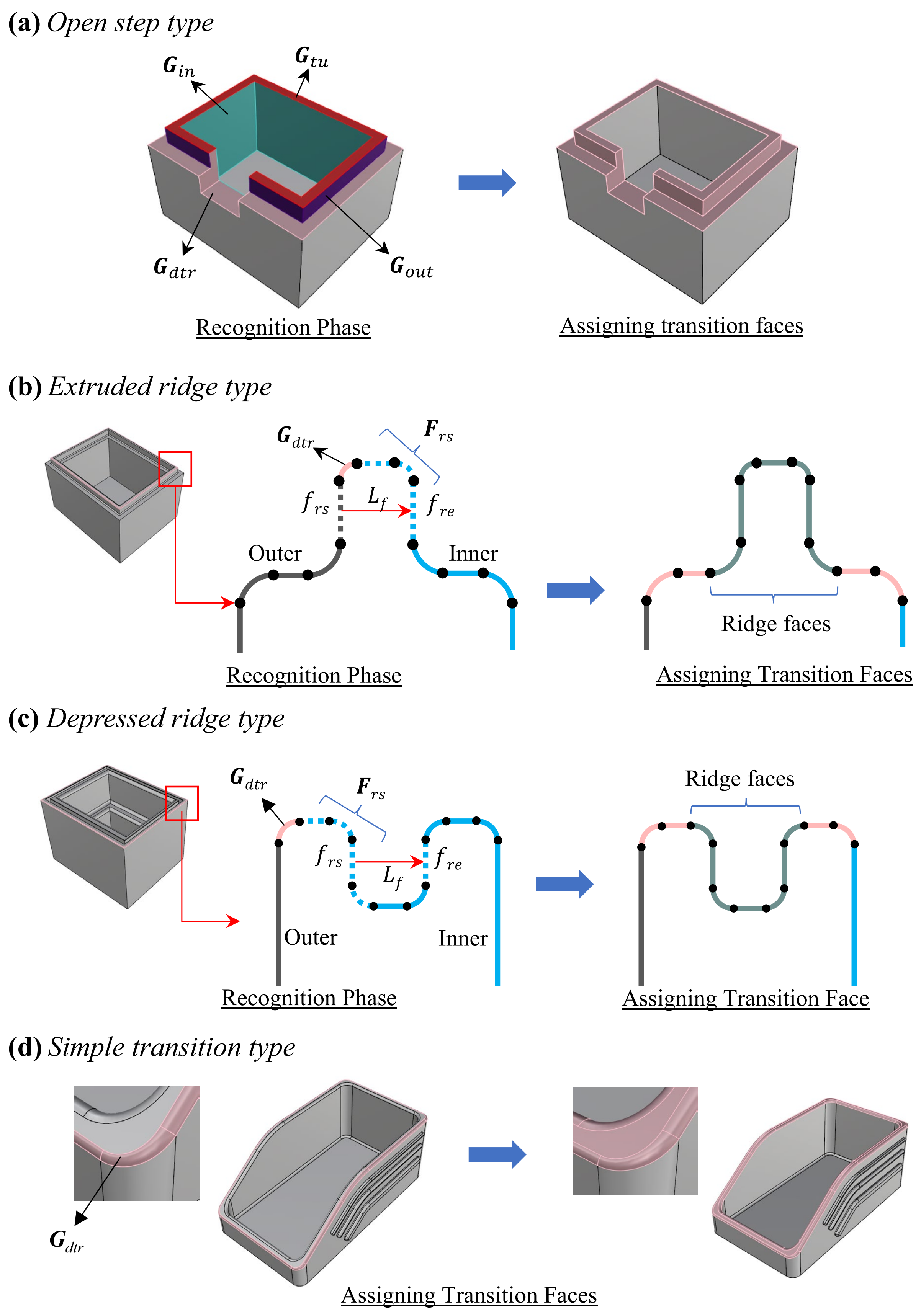


Fig. 8 Immediate results for transition faces recognition, **a** open step, **b** extruded ridge, **c** depressed ridge, and **d** simple transition

its surface normal to intersect with a face f_{re}. The conditions for determining the "depressed ridge" type is as follows:

(a) Check every pair of f_{rs} and f_{re} along the loop of side walls and evaluate a distance for each pair. If all distances are smaller than d_w, then the transition is a "depressed ridge" type.
(b) Otherwise, the transition is not a "depressed ridge" type.

When the transition is a "depressed ridge" type, the transition faces are divided into three regions: (1) ridge faces: the faces between f_{rs} and f_{re} are regarded as ridge faces, where f_{rs} and f_{re} are included. The fillets that are adjacent to f_{rs} and f_{re} are also included as ridge faces; (2) the horizontal face outside and its neighboring fillet; and (3) the horizontal face inside and its neighboring fillet. All three regions of faces can be obtained by using the adjacency relationship of f_{rs} and f_{re}. Figure 8c shows the definition of $\boldsymbol{G}_{dtr}$, $\boldsymbol{F}_{rs}$, f_{rs} and f_{re}, and transition faces for the "depressed ridge" type.

4.2.4 Simple transition

When a model does not belong to any of the above three types, it is regarded as a simple transition. Ideally, a simple transition only involves a face along the thickness direction. However, fillets may exist on one or both sides. When fillets exist, they should be included as transition faces also. Figure 8d shows the definition of $\boldsymbol{G}_{dtr}$ and transition faces for the "simple transition" type.

4.3 Inner face recognition

Inner faces are divided into transition, wall, bottom and protrusion faces. When multiple-layer wall exists, the faces can further be divided into wall and step-wall faces. As transition faces have been recognized, the remaining faces, including wall, bottom, step-wall and protrusion faces are recognized in sequence. When fillets exist, they must be regarded as either wall, bottom or protrusion faces in accordance with the adjacency relationship. When a fillet is adjacent to two different types of face, the priority of the face type that it is assigned is protrusion, bottom and wall face in sequence.

4.3.1 Recognition of wall faces

Wall faces are generally connected to transition faces by convex edges and are perpendicular to the open bounding plane. However, not all wall faces are necessarily connected to transition faces. For example, when a wall face is composed of several faces, only one of them is connected to a transition face, while the others are not. Also, not all wall faces are exactly perpendicular. Some of them may be tilted slightly. Therefore, the procedures for detecting wall faces are described below:

(1) Assign initial wall faces: For all inner faces, except transition faces, the faces that connect to transition faces by convex edges are assigned as initial wall faces $\boldsymbol{G}_{rw}$.
(2) Determine the other wall faces recursively: Consider that f_i is an inner face whose face type hasn't been determined. If it meets the following conditions, then it is regarded as a wall face: (a) f_i is adjacent to a face in $\boldsymbol{G}_{rw}$ by a convex edge, and (b) $\theta_f > \theta_t$, where θ_f is the angle between $\boldsymbol{n}_f$ and $+Z$, n_f is the surface normal of f_i and θ_t is the inclined angle allowed, 10° in this study. This step is repeated for all remaining inner faces recursively. Whenever a face is regarded as a wall face, it is put into $\boldsymbol{G}_{rw}$.
(3) Check wall faces: Some faces may be wrongly recognized as wall faces because a transition face may be a virtual face that crosses over multiple features. The procedures to detect and modify erroneous wall faces are described below:

Consider that f_{w1} is a wall face in $\boldsymbol{G}_{rw}$. Generate a line from the centroid of f_{w1} towards $-\boldsymbol{n}_f$, where $\boldsymbol{n}_f$ is the surface normal. Find a face f_o that intersects with the line and is the closest face of f_{w1}.
If f_o is not an outer face, then f_{w1} is not a wall face. Remove f_{w1} from $\boldsymbol{G}_{rw}$.
If f_o is an outer face, then record the distance d_1 between f_{w1} and f_o.

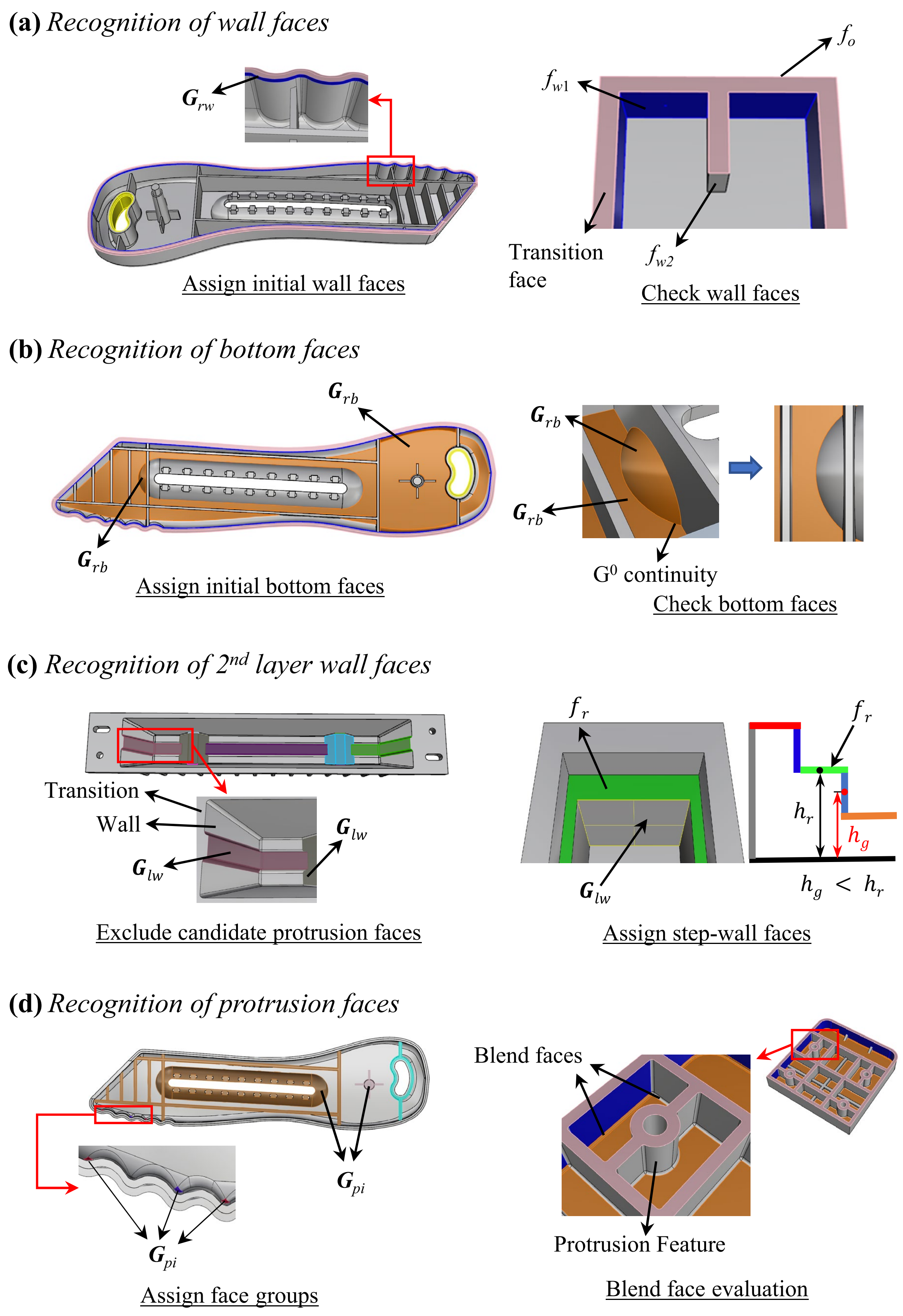
This step is repeated for all faces in $\boldsymbol{G}_{rw}$.

(b) For all wall faces in $\boldsymbol{G}_{rw}$, check if several faces point to the same outer face. Keep the one with the minimum distance in $\boldsymbol{G}_{rw}$, while the others are removed from $\boldsymbol{G}_{rw}$.

All faces in $\boldsymbol{G}_{rw}$ are the final wall faces. Figure 9a shows the results of Procedures (1) and (3) for two examples, respectively.

4.3.2 Recognition of bottom faces

Bottom faces are located on the bottom of the inner faces, mostly characterized by concave edges on the outer loop or connecting with wall faces. Most bottom faces are horizontal, but a small inclined angle is allowed. The procedures for detecting bottom faces are described below:


(a) Recognition of wall faces
G_{rw}
Assign initial wall faces
f_o
f_{w1}
Transition face
f_{w2}
Check wall faces
(b) Recognition of bottom faces
G_{rb}
G_{rb}
Assign initial bottom faces
G_{rb}
G_{rb}
G^0 continuity
Check bottom faces
(c) Recognition of 2nd layer wall faces
Transition
Wall
G_{lw}
G_{lw}
Exclude candidate protrusion faces
f_r
f_r
h_r
h_g
G_{lw}
$h_g < h_r$
Assign step-wall faces
(d) Recognition of protrusion faces
G_{pi}
G_{pi}
Assign face groups
Blend faces
Protrusion Feature
Blend face evaluation

◂**Fig. 9** Immediate results for inner face types recognition, **a** recognition of wall faces, **b** recognition of bottom faces, **c** Recognition of 2nd layer wall faces, and **d** recognition of protrusion faces

(1) Assign initial bottom faces: Consider that f_i is an inner face whose face type hasn't been determined. If f_i meets the following conditions, then it is regarded as an initial bottom face: (a) f_i is not a fillet, (b) all edges on the outer loop are concave, except the edge adjacent to a hole face, and (c) $\theta_f > \theta_t$, where θ_f is the angle between $\boldsymbol{n}_f$ and $+Z$, $\boldsymbol{n}_f$ is the surface normal of f_i and θ_t is the inclined angle allowed, 10° in this study. This step is repeated for all inner faces recursively, except transition and wall faces. Whenever a face is regarded as a bottom face, it is put into $\boldsymbol{G}_{rb}$.
(2) Assign fillets as bottom faces: The fillets that are adjacent to initial bottom faces should be regarded as bottom faces. Consider a face f_i that is adjacent to a vertex on the outer loops of the faces in $\boldsymbol{G}_{rb}$. If f_i meets the following condition, then it is regarded as a bottom face: (a) f_i is a fillet, or (b) f_i is not a fillet, but it connects to a face in $\boldsymbol{G}_{rb}$ convexly. Condition (b) is applied for some tiny faces that are not considered as fillets. All faces that are adjacent to the faces in $\boldsymbol{G}_{rb}$ are checked recursively. Whenever a face is regarded as a bottom face, it is put into $\boldsymbol{G}_{rb}$.
(3) Assign isolated bottom faces separated by ribs: It may occur that some isolated bottom faces are separated by ribs. To overcome this issue, consider an inner face f_i whose face type hasn't been determined. If f_i meets the following conditions, then it is regarded as a bottom face: (a) f_i is a fillet, (b) it is adjacent to a wall face, and (c) it is adjacent to rib faces. Whenever a face is regarded as bottom face, it is put into $\boldsymbol{G}_{rb}$.
(4) Check bottom faces: Some protrusion faces may wrongly be regarded as bottom faces. When it happens, the common edge at two adjacent faces will be concave and G^0 continuous. Therefore, detect two adjacent faces in $\boldsymbol{G}_{rb}$ that are G^0 continuous and connected concavely. Regard the one with a higher centroid along the Z direction as a protrusion face, while the other one with a lower centroid as a bottom face. Update the faces in $\boldsymbol{G}_{rb}$. All neighboring faces in $\boldsymbol{G}_{rb}$ with concave edges should be checked in sequence.

All faces in $\boldsymbol{G}_{rb}$ are the final bottom faces. Figure 9b shows the results of Procedures (1) and (4) for one example.

4.3.3 Recognition of step-wall faces

Till now, transition, wall (1st layer) and bottom faces on inner faces have been recognized. When there is only one layer of wall faces, the wall and bottom faces are directly connected to each other. However, when there are multiple-layer wall faces, some of the faces between wall and bottom faces are still undetermined yet. The undetermined faces may belong to either step-wall or protrusion face. To recognize step-wall faces, a set of reference faces that connect to wall faces are obtained first. The remaining faces are then divided into groups. Protrusion faces are then recognized and excluded from the groups. The final faces in the groups are step-wall faces. Two algorithms are employed to detect protrusion faces. First, most protrusion faces on a group have at least a set of face pair that are parallel or nearly parallel to each other. A face intersection check can be performed to detect this kind of protrusion. Second, some protrusion faces may not be detected by the first algorithm, but they are higher than the neighboring wall faces. A check of the heights can detect his kind of protrusion. The procedures of step-wall face recognition are described as follows:

(1) Assign reference faces f_r: for all inner faces, except transition, wall and bottom faces, the faces that directly connect to wall faces are evaluated, denoted as reference faces f_r. The remaining inner faces that connect convexly, beside f_r, are divided into grouped $\boldsymbol{G}_{lw}$.
(2) Exclude candidate protrusion faces based on face intersection check: the next step is to check all faces in each group $\boldsymbol{G}_{lw}$ and exclude the groups that belong to candidate protrusion faces. Generate a line for every face f_i in $\boldsymbol{G}_{lw}$ towards $-\boldsymbol{n}_f$, where $\boldsymbol{n}_f$ is the surface normal, and check the intersection between the line and the closest face f_c. The conditions for a group in $\boldsymbol{G}_{lw}$ are as follows:

(a) If one of the closest faces f_c is an inner face, then exclude that group of faces from $\boldsymbol{G}_{lw}$.
(b) Otherwise, keep that group in $\boldsymbol{G}_{lw}$.
(c) Exclude candidate protrusion faces based on heights and assign step-wall faces: for all groups of faces in $\boldsymbol{G}_{lw}$, if the faces on a group belong to step-wall faces, then all faces in that group should be lower than the reference faces f_r. On the contrary, if some of the faces are higher than the reference faces f_r, then all faces in that group are considered as protrusion faces. Consider that the maximum height of all centroids of the faces in f_r along the z direction is h_r. Also, the maximum height h_g for all centroids of the faces in a group $\boldsymbol{G}_{lw}$ is evaluated. The faces in $\boldsymbol{G}_{lw}$ are determined as follows:
(d) If $h_g < h_r$, then all faces in that group are step-wall faces.
(e) Otherwise, all faces in that group are not.

Figure 9c shows the results of Procedures (2) and (3) for two examples, respectively.

4.3.4 Recognition of protrusion faces

Inner faces can be divided into transition, wall, bottom and protrusion faces. Once transition, wall and bottom faces are recognized, the remaining faces are grouped in accordance with the adjacency relationship. Most of the groups can be regarded as protrusion faces, but there are still minor groups that must be regarded as either wall or bottom faces. Fillets should be assigned as one face type too. When a fillet is connected to a protrusion face, it is regarded as part of that protrusion group. The procedures are described below:

(1) Assign face groups: All remaining inner faces whose face type hasn't been determined are grouped in accordance with the adjacency relationship. The faces on each group are adjacent to each other. If the faces of a blind hole are adjacent to those of a group, then the faces of this hole are included to the group also. It is noted that some small fillets which haven't been assigned yet will be recognized as individual groups. It results in protrusion groups $\boldsymbol{G}_{pi}$.

(2) Check face intersection on each group: For a face f_i in a group $\boldsymbol{G}_{pi}$, generate a line from the centroid of f_i towards $-\boldsymbol{n}_f$. Find a face f_c that intersects the line and is closest to f_i. It is noted that all faces on the same group must be checked. Based on the type of f_c, some protrusion groups are determined, as follows:

(a) If any intersection face f_c is an inner face or hole face, then regard the faces in $\boldsymbol{G}_{pi}$ as protrusion faces.
(b) Otherwise, proceed to next step.
(c) Determine a flag k for each group: The faces on each of the remaining groups may belong to protrusion, wall or bottom faces depending on the status of a flag k on each group. Check the number of faces on each group first.
(d) If the number of faces is less than 2, then set k as true.
(e) Otherwise, proceed to check the intersection by generating a line along the surface normal for each f_i.
(f) If all f_i intersect with any face, then set k as true.
(g) If there is one f_i on the group that does not intersect any face, then check the neighboring face of f_i. If a bottom face is the neighboring face, then set k as true. Otherwise, set k as false.
(h) Check the remaining face groups: Based on the status of k, the face type for each group are determined.
(i) If k is false, then assign all faces on the group as protrusion faces.
(j) Otherwise, check neighboring faces of the group.
(k) If the group has a wall face as its neighboring face, then sort some of the faces on the group to be either wall or bottom. Vertical faces will be assigned as wall faces, while the remaining faces will be bottom faces.
(l) If there is no wall face as its face neighboring, then convert faces on the group as bottom faces.

Figure 9d shows the results of Procedures (1) and (4) for two examples, respectively.

4.4 Outer faces recognition

Figure 10 shows the flowchart of the face types evaluation for outer faces, where the inputs are the holes, fillets, inner faces and parting line, and the outputs are the composition of faces on the outer faces. The faces are initially divided into outer wall and bottom faces. The model is then divided into concave and non-concave bottom types. For the non-concave bottom type, some of the transition, outer bottom and outer wall faces are modified in accordance with the adjacency relationship. The flange faces are then recognized. Outer wall and bottom faces are finally determined. The model is then divided into concave and non-concave wall types for evaluating protrusion faces. For the concave bottom type, outer wall and bottom faces are modified first. It follows the recognition of flange faces. The remaining procedure is the same as that of non-concave bottom type. A detailed description of the procedures is shown below.

4.4.1 Initially separate outer wall and bottom faces

Outer wall and bottom faces are initially evaluated by checking the intersection with the boundary box and inner faces. The procedures are as follows:

(1) Set the outer faces that are adjacent to the parting line as outer wall faces: most of the faces that are adjacent to the parting line are vertical or nearly vertical, and hence can be considered as outer wall faces.
(2) Separate the outer faces into two regions by θ_f: consider an outer face f_i with an angle θ_f between $\boldsymbol{n}_f$ and $+Z$. It is assigned to one of the following two regions: (1) Region I: $0° \le \theta_f \le 90° + \varepsilon$, and (2) Region II: $90° + \varepsilon < \theta_f \le 180°$, where ε is a draft angle, 3° in this study.

When a face f_i is in Region I, the outer wall and bottom faces are determined by projecting a line from the target face and check the number of intersections. Different face type has different number of intersections. Compute a parameter C_5, where C_5 denotes the status of the intersection. Project a line from the centroid of f_i towards $\pm X$, $\pm Y$ and $\pm Z$, respectively, and check if it intersects any face on the model. If an intersection occurs, C_5 is increased by 1 (The values of C_5:

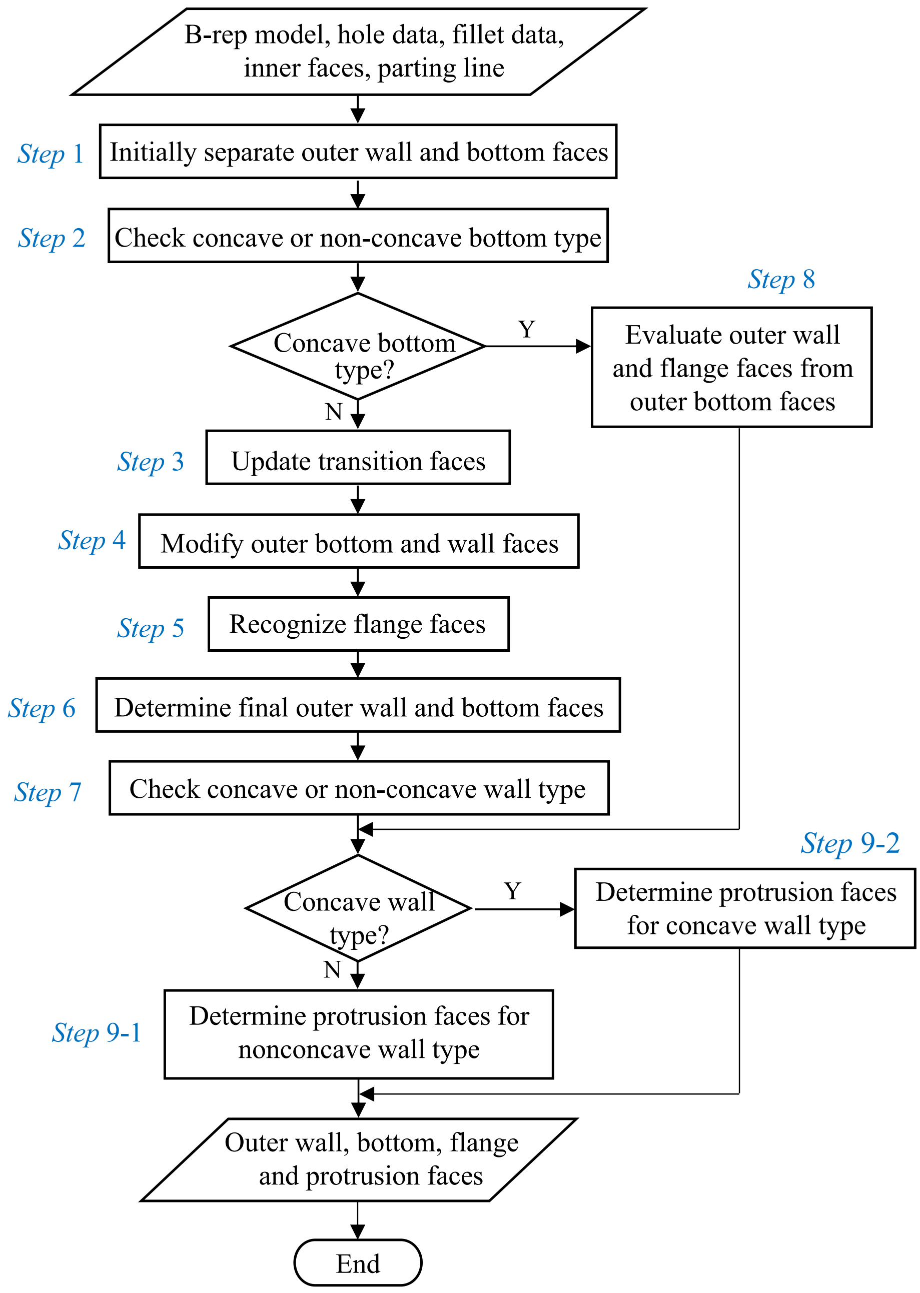

B-rep model, hole data, fillet data, inner faces, parting line
Step 1
Initially separate outer wall and bottom faces
Step 2
Check concave or non-concave bottom type
Concave bottom type?
Y
N
Step 8
Evaluate outer wall and flange faces from outer bottom faces
Step 3
Update transition faces
Step 4
Modify outer bottom and wall faces
Step 5
Recognize flange faces
Step 6
Determine final outer wall and bottom faces
Step 7
Check concave or non-concave wall type
Concave wall type?
Y
N
Step 9-2
Determine protrusion faces for concave wall type
Step 9-1
Determine protrusion faces for nonconcave wall type
Outer wall, bottom, flange and protrusion faces
End

Fig. 10 Flowchart of the face types evaluation for outer faces

0 ~ 5). Rules of determining whether f_i is an outer wall or bottom face is as follow:

(1) if $C_5 = 5$, then regard f_i as an outer bottom face.
(2) Otherwise, regard f_i as an outer wall face.

The left plot in Fig. 11a shows two faces that are assigned as outer bottom ($C_5 = 4$) and outer wall ($C_5 = 3$), respectively.

When a face f_i is in Region II, the outer wall and bottom faces are determined in a way like that of Region I. Compute a parameter C_4, where C_4 denotes the status of the intersection. Project a line from the centroid of f_i along $\pm X$ and $\pm Y$, respectively, and check if it intersects any inner face on the model. If an intersection occurs, C_4 is increased by 1 (The values of C_4: 0 ~ 4). Rules of determining whether f_i is an outer wall or bottom face is as follow:

(1) If $C_4 = 0$, then regard f_i as an outer bottom face.
(2) If $C_4 = 1$, then
(3) If $\theta_f > 170°$, then regard f_i as an outer bottom face.
(4) Otherwise, regard f_i as an outer wall face.
(5) If $C_4 = 2$ to 4, then
(6) If $C_5 = 5$, then regard f_i as an outer bottom face.
(7) Otherwise, regard f_i as an outer wall face.

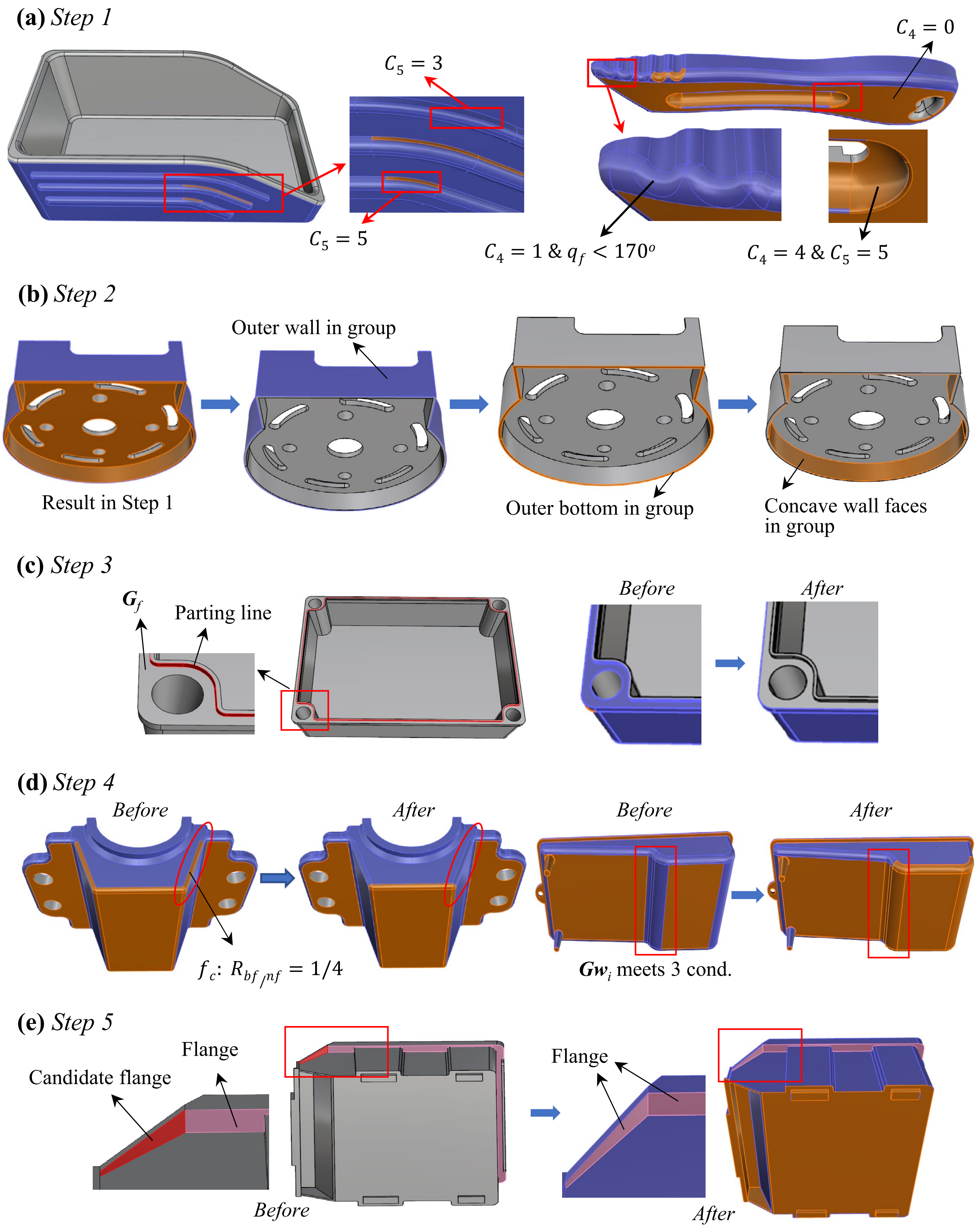


Fig. 11 Immediate results for outer face types recognition, **a** Step 1, **b** Step 2, **c** Step 3, **d** Step 4, and **e** Step 5

The right plot in Fig. 11a shows three faces that are assigned as outer bottom (($C_4 = 0$) and ($C_4 = 4$ amd $C_5 = 5$)) and outer wall ($C_4 = 1$ and $\theta_f < 170°$), respectively.

4.4.2 Check concave or non-concave bottom type

A model is divided into concave or non-concave bottom type. For the concave bottom type, the bottom face is sunken compared with its neighboring wall faces; while for the non-concave bottom type, the bottom face is convexly connected to its neighboring wall faces. Therefore, non-concave bottom faces are normally facing down and can form an individual loop, while concave bottom faces can further be divided into two types of faces: concave bottom and concave wall faces. The procedures for detecting concave and non-concave bottom types are as follows:

(1) Group outer wall faces: Start from an outer wall face that is adjacent to the parting line, find all outer wall faces that are adjacent to each other. Put these faces into a group $\boldsymbol{G}_{ow}$.
(2) Group outer bottom faces and determine non-concave bottom type: An outer bottom face f_i is put into a group $\boldsymbol{G}_{cb}$ if it meets the following two conditions: (1) $\theta_f > 170°$, and (2) f_i is connected to any face in $\boldsymbol{G}_{ow}$. The remaining outer bottom faces are put into a group $\boldsymbol{G}_{ncb}$. Check the boundary loops of the faces in $\boldsymbol{G}_{cb}$.
(3) If there is only a loop, then the model is regarded as a non-concave bottom type. Stop the process.
(4) On the contrary, if there is more than one loop, then the faces in $\boldsymbol{G}_{cb}$ are called concave bottom faces. Proceed Step 3.
(5) Determine concave bottom type: Find an outer bottom face f_i that is in $\boldsymbol{G}_{ncb}$ and is adjacent to a face in $\boldsymbol{G}_{cb}$. Generate a line L_i at the centroid of f_i and along $-\boldsymbol{n}_f$. If L_i intersects an outer wall face in $\boldsymbol{G}_{ow}$, then put f_i into a group $\boldsymbol{G}_{cw}$. Start from f_i, keep checking the other faces in $\boldsymbol{G}_{ncb}$ by neighboring until L_i does not intersect any face in $\boldsymbol{G}_{ow}$. Put all faces with L_i intersecting a face in $\boldsymbol{G}_{ow}$ into $\boldsymbol{G}_{cw}$. Once all faces in $\boldsymbol{G}_{ncb}$ are tested, check the faces in $\boldsymbol{G}_{cw}$. If $\boldsymbol{G}_{cw}$ is not empty, then the model is regarded as a concave bottom type, and the faces in $\boldsymbol{G}_{cw}$ are called concave wall faces. Otherwise, the model is regarded as a non-concave bottom type.

Figure 11b shows the results of outer wall grouping, outer bottom grouping and concave wall evaluation for a model of concave bottom type.

4.4.3 For nonconcave bottom type

4.4.3.1 (A) Step 3: Update transition faces In general, a parting line separates the faces into inner and outer faces. The first layer of inner faces that are adjacent to the part line are transition faces, whereas the first layer of outer faces that are adjacent to the pat line are outer wall faces. The first layer of outer wall faces is typically vertical or nearly vertical. However, it may occur that the first layer of outer wall faces is close to horizontal. In such a situation, the parting line should be moved outward to cover this layer of faces as transition faces. The procedures to detect such a situation and update transition faces are as follows:

(1) Get faces that are adjacent to the parting line: The outer faces that are adjacent to the parting line are put into a group $\boldsymbol{G}_f$.
(2) Get outer wall faces that are adjacent to the faces in $\boldsymbol{G}_f$: Let f_i be an outer face that is adjacent to a face in $\boldsymbol{G}_f$. If f_i meets the following two conditions, then it is regarded as a transition face:
(3) f_i and all faces in $\boldsymbol{G}_f$ are outer wall faces.
(4) f_i is adjacent to all faces in $\boldsymbol{G}_f$.

When f_i is changed into a transition face, all faces in $\boldsymbol{G}_f$ are also changed into transition faces. The left plot in Fig. 11c indicates the parting line and $\boldsymbol{G}_f$ for an example, while the right plots indicate the situation of outer wall faces before and after the modification.

4.4.3.2 (B) Step 4: Modify outer bottom and wall faces Some of the outer bottom and wall faces may wrongly be recognized. They are detected and modified based on geometric and face adjacency criteria. The procedures are as follows:

(1) Group outer bottom faces $\boldsymbol{GB}_i$: All outer bottom faces that are adjacent to each other are regarded as a group, yielding $\boldsymbol{GB}_i$. The group with the maximum area is denoted $\mathbf{Max}(\boldsymbol{GB}_i)$.
(2) Check and modify outer bottom faces: For a face f_i in $\mathbf{Max}(\boldsymbol{GB}_i)$, compute two parameters $R_{bf/nf}$ and $L_{w/all}$, where the former denotes the ratio between the number of outer bottom faces neighboring to f_i and the number of faces neighboring to f_i, and the latter denotes the ratio between the length of edges neighboring to outer wall faces and the length of all edges neighboring to f_i. The face type for f_i is determined in accordance with the following conditions:
(3) If $\theta_f > 170°$, f_i is kept no change.
(4) If $\theta_f \leq 170°$,
(5) If $R_{bf/nf} > 0.5$, f_i is kept no change.
(6) If $R_{bf/nf} < 0.5$, f_i is modified as an outer wall face.
(7) If $R_{bf/nf} = 0.5$, then if $L_{w/all} < 0.5$, then f_i is kept no change. Otherwise, f_i is modified as an outer wall face.

(8) Group outer wall faces $\boldsymbol{GW}_i$: All outer wall faces that are adjacent to each other are regarded as a group, yielding $\boldsymbol{GW}_i$.

(9) Check and modify outer wall faces: For an outer wall face f_i that is adjacent to the faces in $\mathbf{Max}(\boldsymbol{GB}_i)$, check if it meets the following conditions:

(10) f_i is not a fillet.

(11) f_i is not adjacent to the parting line.

(12) A line at the centroid of f_i and along $\boldsymbol{n}_f$ intersects the boundary plane on –*Z*.

If yes, then regard f_i as an outer bottom face and put it into $\mathbf{Max}(\boldsymbol{GB}_i)$. On the contrary, if no, then stop the search along f_i.

(E) Regroup outer bottom and wall faces: Outer bottom and wall faces that are adjacent to each other are respectively grouped again.

The left example in Fig. 11d shows a situation in Procedure (2), where outer bottom is modified as outer wall; while the right example in Fig. 11d shows a situation in Procedure (4), where outer wall is modified as outer bottom.

4.4.3.3 (C) Step 5: Recognize flange faces Search flange faces both from transition faces and outer bottom faces. Most flange faces are parallel to transition faces, and hence can be evaluated by checking the intersection of lines projected from transition faces. However, not all flange faces can be obtained. Therefore, lines projected from outer bottom faces are also checked, which can yield the residual flange faces. The procedures are as follows:

(1) Evaluate candidate flange and flange faces using transition faces: Generate one or two lines along − Z direction on each transition face and find faces that intersect with the lines. If a transition face has at most 4 edges, apply one line for the intersection check. Otherwise (i.e. more than 4 edges), apply two lines for the intersection. For each transition face f_{ti}, a line L_i from a face point ($\boldsymbol{P}_i$) along –Z is generated. If there are two face points, then two lines will be generated. The faces that intersect with any of the lines are obtained. The one with the shortest distance is regarded as f_c. Define two parameters d_z and d_t, where the former denotes 0.5 length of the boundary box along *Z* direction, and the latter denotes the shortest distance between f_{ti} and f_c (along − *Z* direction). If f_c meets the following criteria, then it is regarded as a flange face:

(a) f_c is an outer face

(b) $d_t < d_z$

(c) $\theta_f \geq 170°$ (i.e. f_c is almost facing down).

In addition, if f_c meets the following criteria, then it is regarded as a candidate flange face:

(a) f_c is an outer face

(b) $d_t < d_z$

(c) $100° < \theta_f < 170°$ (i.e. f_c is an inclined face).

Because f_c is an inclined face, further rules must be applied later to determine if f_c is a flange face.

(B) Evaluate candidate flange and flange faces using outer bottom faces: For some cases, applying transition faces only cannot obtain all flange faces. Outer bottom faces are mostly facing down, and flange faces also fit this feature. Therefore, a line from an outer bottom face is also tested to check if it can intersect any transition face. If an intersection occurs, the outer bottom face is also regarded as a flange face. For each outer bottom face f_{bi}, a line L_i from a face point ($\boldsymbol{P}_i$) along + *Z* is generated. If there are two face points, then two lines will be generated. The faces that intersect with any of the lines are obtained. The one with the shortest distance is regarded as f_c. Define two parameters d_z and d_b, where the former denotes 0.5 length of the boundary box along *Z* direction, and the latter denotes the shortest distance between f_{bi} and f_c (along + *Z*). If f_c meets the following criteria, then it is regarded as a flange face:

(C) f_c is an outer bottom face (not in $\boldsymbol{Max}(\boldsymbol{GB}_i)$)

(D) $d_b < d_z$

(E) $\theta_f \geq 170°$ (i.e. f_c is almost facing down).

In addition, if f_c meets the following criteria, then it is regarded as a candidate flange face:

(a) f_c is an outer bottom face (not in $\boldsymbol{Max}(\boldsymbol{GB}_i)$)

(b) $d_b < d_z$

(c) $100° < \theta_f < 170°$ (i.e. f_c is an inclined face).

(d) Group candidate flange faces: In Steps 1 and 2, some faces are already regarded as flange faces. However, some other faces are regarded as candidate flange faces. All candidate flange faces that are adjacent to each other are regarded as a group, yielding $\boldsymbol{GF}_i$.

(e) Modify candidate flange faces: If any candidate flange face in a group $\boldsymbol{GF}_i$ is adjacent to a flange face, then all faces in that group are changed into flange faces. Otherwise, all faces in that group are returned to the original face type (i.e. either outer wall or bottom face type)

(f) Regroup outer wall, bottom and flange faces: All types of faces are regrouped again. It yields $\boldsymbol{GW}_i$, $\boldsymbol{GB}_i$ and $\boldsymbol{GF}_i$.

The example in Fig. 11e shows two faces that are initially detected as flange and candidate flange faces, respectively. The candidate flange face is finally modified as a flange face as it is adjacent to a flange face.

4.4.3.4 (D) Step 6: Determine final outer wall and bottom faces All outer wall and bottom faces are divided into separate groups now. However, some of the face types are still wrong and must be corrected. The procedures are as follows:

(1) Compute $\mathbf{Max}(\boldsymbol{GB}_i)$, where the number of outer bottom faces is the largest: Some of the outer wall faces will separate outer bottom faces into different groups. If all faces in a group $\boldsymbol{GW}_i$ do not connect with any flange or transition face, then all faces in $\boldsymbol{GW}_i$ are changed to outer bottom faces. All outer bottom faces are re-grouped again. The one with the largest number of faces is called $\mathbf{Max}(\boldsymbol{GB}_i)$.
(2) Check outer bottom faces: There are several groups of outer bottom faces $\boldsymbol{GB}_i$. Keep the faces in $\boldsymbol{Max}(\boldsymbol{GB}_i)$ as outer bottom faces, whereas the faces on the other groups are changed to outer wall faces.
(3) Check outer wall faces: All outer wall faces in a group $\boldsymbol{GW}_i$ are checked.
(4) If any of the faces in $\boldsymbol{GW}_i$ is adjacent to both flange and transition faces, then all faces in $\boldsymbol{GW}_i$ are regarded as flange faces.
(5) If all faces in $\boldsymbol{GW}_i$ are not adjacent to any flange face, transition face or bottom face, then all faces in $\boldsymbol{GW}_i$ are regarded as outer bottom faces.

Figure 12a shows some erroneous bottom and wall faces detected in this step and the correction of them.

4.4.3.5 (E) Step 7: Check concave or non-concave wall type Till now, outer protrusion faces are regarded as either outer wall, bottom or flange faces. The model is further classified as two types for the recognition of protrusion faces. For the concave wall type, substantial protrusions exist on outer faces and divide wall faces into many regions. It looks like many concave regions exist on outer faces. On the contrary, for the non-concave wall type, protrusions may exist on outer faces, but are distributed individually. Most of the outer wall faces are not divided.

During the recognition of inner and outer faces, there is a stage that generates several groups of potential inner faces. Only one group of faces is finally regarded as inner faces, while the other groups are located on outer faces, which are called candidate inner faces $\boldsymbol{G}_{cif}$ here. Consider that f_i is a face in $\boldsymbol{G}_{cif}$ and n_{cif} is the intersection count for faces in $\boldsymbol{G}_{cif}$. Project a line from the centroid of each f_i along its $\boldsymbol{n}_f$. If it intersects with any face in $\boldsymbol{G}_{cif}$ then n_{cif} is increased by 1. All groups of candidate inner faces $\boldsymbol{G}_{cif}$ are checked one by one. If any group meets the following criteria, then the model is regarded as a “concave wall type”:

(1) There exists a face f_c that is nearly horizontal and facing up, i.e. $n_{cif} \le \varepsilon$, where ε is the draft angle.
(2) At least three faces intersect with other faces on the same group, i.e. $n_{cif} > 2$.

If all groups $\boldsymbol{G}_{cif}$ do not meet the above criteria, then the model is regarded as a “non-concave wall type”. The left plot in Fig. 12b shows an example of non-concave wall type. The right plot in Fig. 12b shows an example with 33 groups of candidate inner faces in $\boldsymbol{G}_{cif}$. This example is regarded as a concave wall type as some of the groups meet the above-mentioned conditions.

4.4.4 For concave bottom type

The difference between concave bottom and non-concave bottom is that there are two layers of outer wall for the former, while there is only one layer of outer wall for the latter. The concave wall separates outer bottom faces into two regions, where the first region has been detected in Step 2, while the second region has not. Figure 12c shows an example of concave bottom type, where the left and middle plots indicate the results of Steps 1 and 2, respectively. As the middle plot indicates, the 1st region outer bottom is just a simple face, while the 2nd region outer bottom is a complex structure. The faces on the 2nd region must be analyzed again to separate outer wall, outer bottom and flange faces. The algorithm is similar to those used in Steps 1, 4 and 5, and is not addressed again. The right plot in Fig. 12c shows the results of outer wall, outer bottom and flanged faces obtained for the 2$^{\text{nd}}$ region of outer bottom faces.

4.4.5 Determine protrusion faces

4.4.5.1 Step 9–1: For concave wall type Extract protrusion faces from outer faces, including bottom, wall and flange faces. The procedures are as follows:

(1) Extract protrusion faces from outer faces: For each outer face f_i, a line L_i along $-\boldsymbol{n}_f$ is generated, where $\boldsymbol{n}_f$ denotes the surface normal of f_i. The faces that intersect with the line are obtained and the one that has the shortest distance is regarded as f_c. If f_c is an outer face, then regarded f_i as a protrusion face. Otherwise, f_i is kept no change.

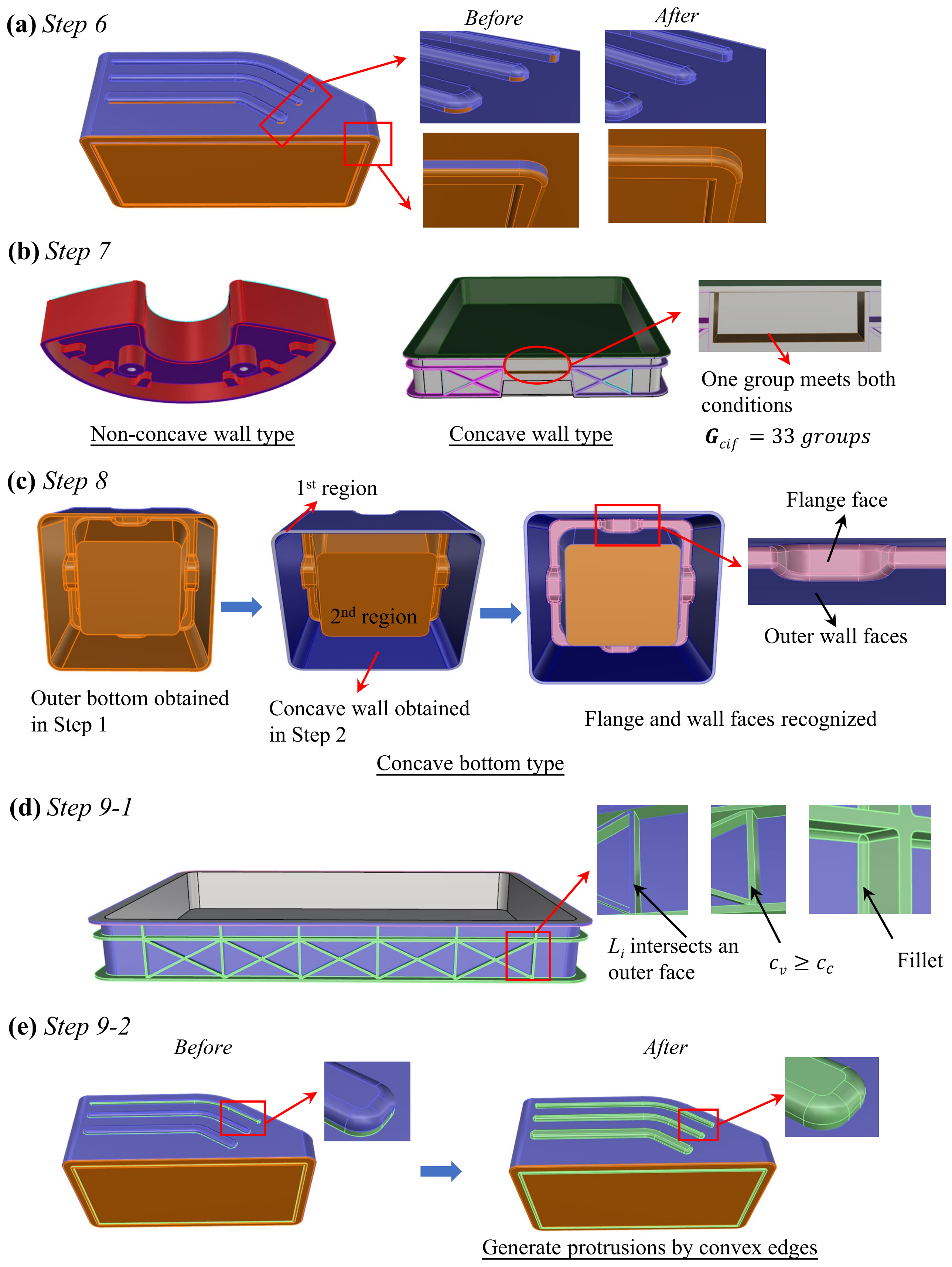


Fig. 12 Immediate results for outer faces recognition, **a** Step 6, **b** Step 7, **c** Step 8, **d** Step 9–1, and **e** Step 9–2

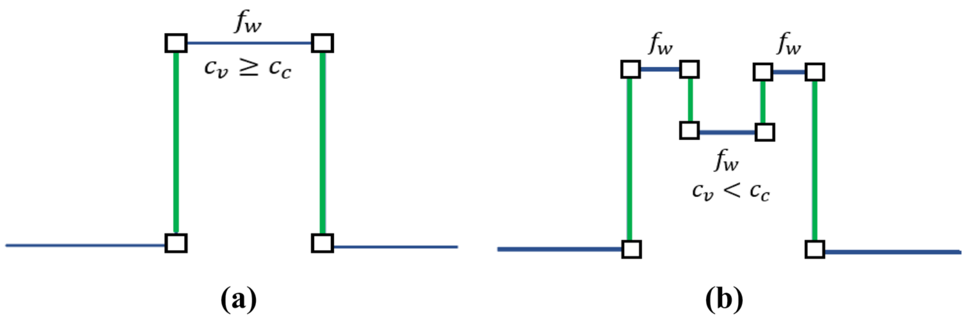


Fig. 13 Two types of protrusion on outer wall face, **a** single-layer protrusion, and **b** multi-layer protrusion

(2) Check and modify outer wall faces: In previous step, some outer wall faces that are on the top of protrusion faces have not been recognized correctly. The convexity of the neighboring edges can be used to check the correctness of these faces. Consider an outer wall face f_w. Define two parameters c_v as the number of convex edges between f_w and its adjacent protrusion faces, and c_c as the number of concave edges between f_w and its adjacent protrusion faces. If f_w is on the top of a protrusion, c_v should be larger than c_c. Otherwise, c_v should be smaller than c_c. That is
(3) If $c_v \geq c_c$, then change f_w as an outer protrusion face.
(4) Otherwise, f_w is kept no change.
(5) Check and modify fillets: When a fillet exists between a protrusion and other types of faces, it is regarded as a protrusion face. Consider a fillet face f_f. Define four parameters c_p, c_w, l_p and l_w as follows: (1) c_p is the number of protrusions faces that are adjacent to f_f, (2) c_w is the number of outer wall faces that are adjacent to f_f, (3) l_p is the total length of the edges of protrusions that are adjacent to f_f, and (4) l_w is the total length of the edges of outer wall faces that are adjacent to f_f. The fillet face f_f is determined according to the following rules:
(6) If $c_p < c_w$, then f_f is kept no change.
(7) If $c_p > c_w$, then change f_f as an outer protrusion face.
(8) If $c_p = c_w$
(9) If $l_p < l_w$, then f_f is kept no change.
(10) If $l_p \geq l_w$, then change f_f as an outer protrusion face
(11) Check and modify protrusion faces: All outer protrusion faces that are adjacent to each other are regarded as a group, yielding $\boldsymbol{GP}_i$. Each group needs to be checked as follows: Define a parameter c_i as follows. Consider a protrusion face f_i in $\boldsymbol{GP}_i$. A line L_i along $-\boldsymbol{n}_f$ is generated. If this line intersects with any protrusion face in $\boldsymbol{GP}_i$, then the flag c_i is true, otherwise c_i is false.
(12) If c_i is true, then all faces in $\boldsymbol{GP}_i$ are kept no change.
(13) If c_i is false, then change all faces in $\boldsymbol{GP}_i$ as outer bottom faces

The three plots highlighted in Fig. 12d show a situation in Procedures (1), (2) and (3), respectively.

Step 9–1 is a stage for detecting protrusion features in concave wall type where the protrusion features cover part of the outer wall. Concave wall types are classified based on the presence of a protrusion on the outer wall, which serves as a reinforcement for the model. In step 9–1(2), two parameters, c_v and c_c, are used to analyze the faces on the top of the protrusion feature. Generally, a single-layer protrusion formed in concave wall type is $c_v \geq c_c$, as shown in Fig. 13a. However, for multi-layer protrusions, there could exist concave faces with $c_v < c_c$, such as the case in Fig. 13b. We have not found any concave protrusion on the concave wall type for the test cases we used. However, Step 9–1(2) must be modified if multi-layer protrusions exist on concave wall type.

4.4.5.2 Step 9–2: For non-concave wall type

Extract protrusion faces from outer faces, including bottom, wall and flange faces.

(1) Extract candidate protrusion faces from outer faces: For each outer face or outer hole face f_i, a line L_i along $-\boldsymbol{n}_f$ is generated, where $\boldsymbol{n}_f$ denotes the surface normal of f_i. The faces that intersect with L_i are obtained. The one with the shortest distance is regarded as f_c. Put f_i in different stacks in accordance with the following conditions:

(a) If f_i is an outer face and is not adjacent to any transition face
(b) If f_c is an outer face, then put f_i into the stack $\boldsymbol{G}_1$.
(c) If f_c and f_i are equal, then put f_i into the stack $\boldsymbol{G}_1$.
(d) If f_i is a hole face and f_c is an outer face, then put f_i into the stack $\boldsymbol{G}_2$.
(c) Otherwise, skip f_i.

The faces in each stack will further be checked next.

(B) Extract protrusion faces from candidate protrusion faces: For each face f_i in $\boldsymbol{G}_1$, generate a line L_i along $-\boldsymbol{n}_f$. If L_i intersects with any face in $\boldsymbol{G}_i$, then regard all faces in $\boldsymbol{G}_1$ as protrusion faces. Perform the same check for all faces in $\boldsymbol{G}_2$ too.

(C) Generate protrusion faces by convex edges: Start from a face f_i that is adjacent to a protrusion face, but not a protrusion face. Search its neighboring faces that are convexly connected. Continue this process until no more face is found. All these faces are regarded as outer protrusion faces.

The example in Fig. 12e shows a situation in Procedure (3), where erroneous outer wall faces are modified as protrusion faces.

5 Results and discussion

A program, written in C + + and based on the Rhino CAD platform and the openNURBS functions, was implemented to test the feasibility of the proposed inner and outer faces recognition algorithm for thin-shell parts. The input data is a B-rep model of the part. The program will recognize fillets and holes first, and then recognize inner and outer faces, transition faces, inner face types, and outer face types in sequence. As the dimension of the parts may be different, three parameters $l_{\max}$, r_{max} and ε in the proposed algorithm are provided for adjusting. The parameter $l_{\max}$ denotes the maximum perimeter of the hole or pocket that will be recognized. The parameter $r_{\max}$ denotes the maximum radius of the fillets. And, the parameter ε denotes the draft angle. The default values for $l_{\max}$, $\theta_{\max}$ and $\varepsilon$are 100 mm, 6.5 mm and 3°, respectively, in this study.

The results of the proposed method for 25 thin-shell parts are divided into four groups, as shown in Figs. 14, 15, 16, 17 respectively, for discussion. In each part, four intermediate results are displayed, including separation of inner and outer faces, transition faces, wall, bottom and protrusion faces on the inside, and wall, bottom and protrusion faces on the outside. Figure 14 shows seven of the parts that are complex on the internal structure, while simple on the outside. Most of the ribs on the inside are connected together, and may even connect to other tubes or bosses (e.g. Cases 1–5). By isolating internal wall and bottom faces first, the remaining protrusion faces can easily be divided into groups. A protrusion classification algorithm can later be implemented to recognize different types of protrusion, e.g. ribs, tubes, columns, and symmetric extrusions [22]. In Cases 6 and 7, substantial through holes exist on bottom faces. If some of the through holes are not recognized, it may affect the recognition of bottom faces. In particular, in Case 6, several holes are lying across multiple faces. It needs the introduction of virtual loop (edges on a loop are G^1 continuous) [39] in order to recognize the loops that cross multiple faces, and hence the corresponding holes.

Figure 15 shows six of the parts that are freeform on the shape or complex on translation and wall faces. In Cases 8 and 9, substantial freeform surfaces exist on the wall or bottom faces. When freeform surfaces exist on the wall, they are usually not perpendicular. This inclination cannot be compensated by the draft angle ε. Therefore, in Sect. 4.3.1, an allowable inclined angle θ_t, 10° in this study, is employed to compensate for the inclination of freeform surfaces. When the inclined angle of a freeform surface is larger than θ_t, it may become difficult to distinguish from the wall and bottom faces, which is considered as another type of thin-shell part and will be discussed elsewhere. In Cases 10 and 11, the transition faces belong to “simple translation” although some of the faces are inclined along the X or Y direction. The faces on each side along the thickness direction are composed of three faces, a main face and two fillets that connect to the main face. It is noted that if the proposed algorithm is not employed, part of the fillets may easily be regarded as wall faces, instead of transition faces, In Case 12, substantial through holes exist on the bottom faces. In particular, several holes are lying on more than three faces, including a fillet. When these holes are not recognized correctly, some of the bottom faces would not be detected correctly. In addition, the fillet on the outer wall is difficult to deal with as it has a large inclined angle. This fillet is correctly recognized as an outer wall face here. Case 13 should be regarded as one of the most difficult cases as it has a very complex wall structure and many through holes of irregular types. It needs the recognition of multi-virtual loops (edges on a loop are G^0 continuous) [39] in order to recognize all irregular holes correctly. The results also show that all inner and outer face types are recognized correctly.

Figure 16 shows seven of the parts that are complex on the external structure. Cases 14 to 16 show the examples of nonconcave wall type, where protrusion faces are sparsely distributed. In Cases 14 and 15, protrusion faces exist on the inside too. In Case 16, flange faces exist on the reverse side of transition faces. All three examples indicate that all face types on the inside and outside are recognized correctly. Cases 17 and 18 show the examples of concave wall type. All faces on the outer faces are carefully checked, especially protrusion faces colored in green and fillets that exist between different types of faces. The result shows that all protrusion faces and fillets on the outside are correctly recognized. Cases 19 and 20 show two examples of concave bottom type. The outer bottom faces on both cases are sunken compared with their neighboring outer wall faces. In Case 19, some protrusion faces are further extracted from outer bottom faces; while in Case 20, some flange faces are

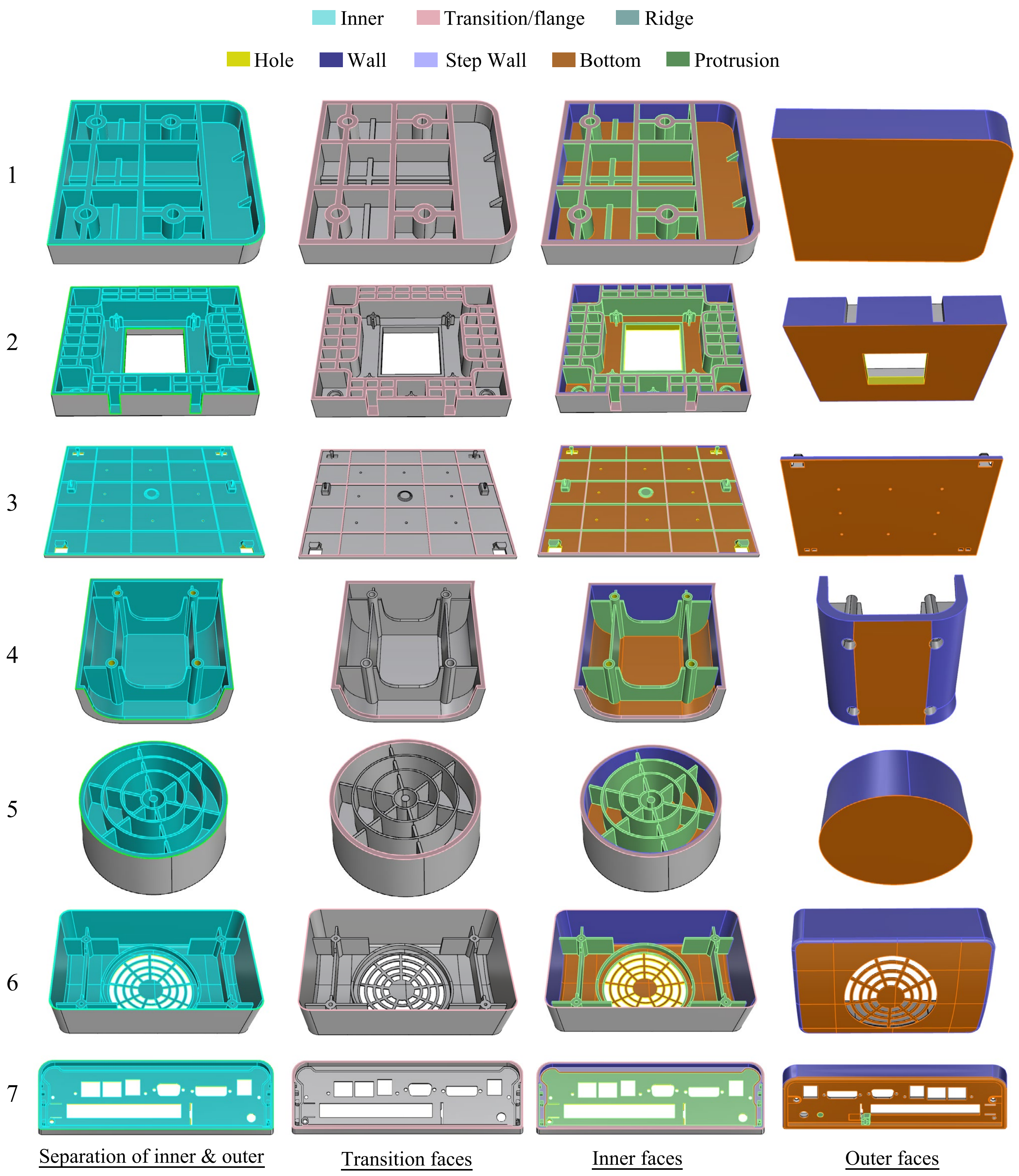


Fig. 14 Results of inner and outer faces recognition- complex structure outside

further extracted from outer bottom faces. All these results are achieved automatically by the program.

Figure 17 shows five of the parts with different types of transition faces, other than "simple transition". The transition faces in Cases 21 to 25 are "depressed ridge", "simple translation with holes", "extruded ridge", "step", and "open step", respectively. All these examples show that the proposed algorithm can detect different types of

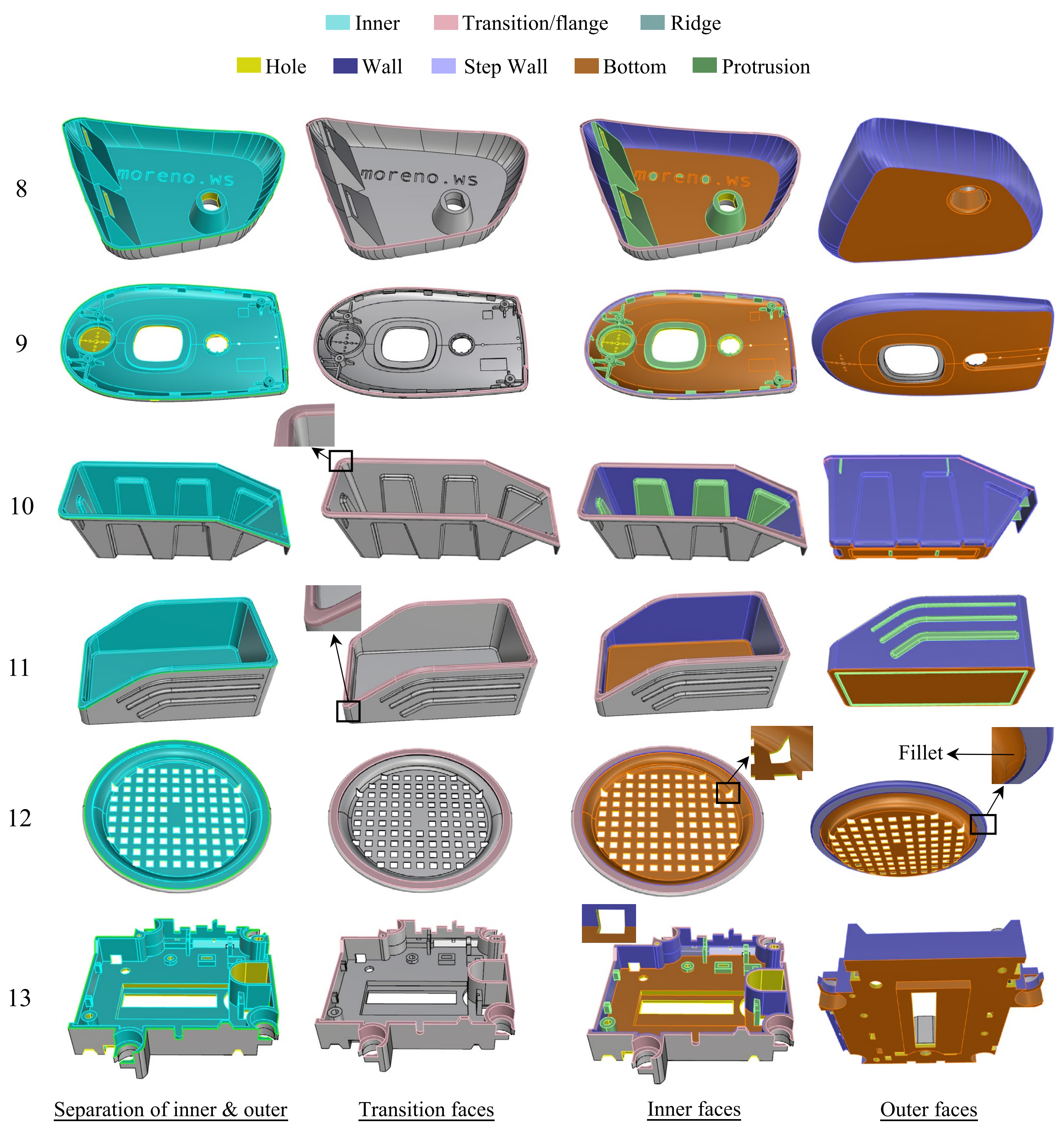


Fig. 15 Results of inner and outer faces recognition- freeform surfaces and complex shape on translation and wall faces

translation faces accurately. Correct classification and recognition of transition faces is very important as the remaining inner and outer faces recognition all counts on the accuracy of translation faces. In addition, the inner wall faces are divided into multiple layers in Cases 22 to 24, where the wall and step-wall faces are colored in blue and light blue, respectively. Cases 22 and 23 show the typical situation on which the wall and bottom faces are separated by step-wall faces. Fillets existing between wall and step-wall faces can also be detected correctly. However, Case 24 shows a particular situation that part of the wall faces are connected to the bottom face directly. All these results show that different kinds of multi-layer wall faces can be recognized satisfactorily.

Table 1 summarizes the results of inner and outer faces recognition for 25 cases, where the number of total faces, through holes, inner faces and outer faces are listed to check

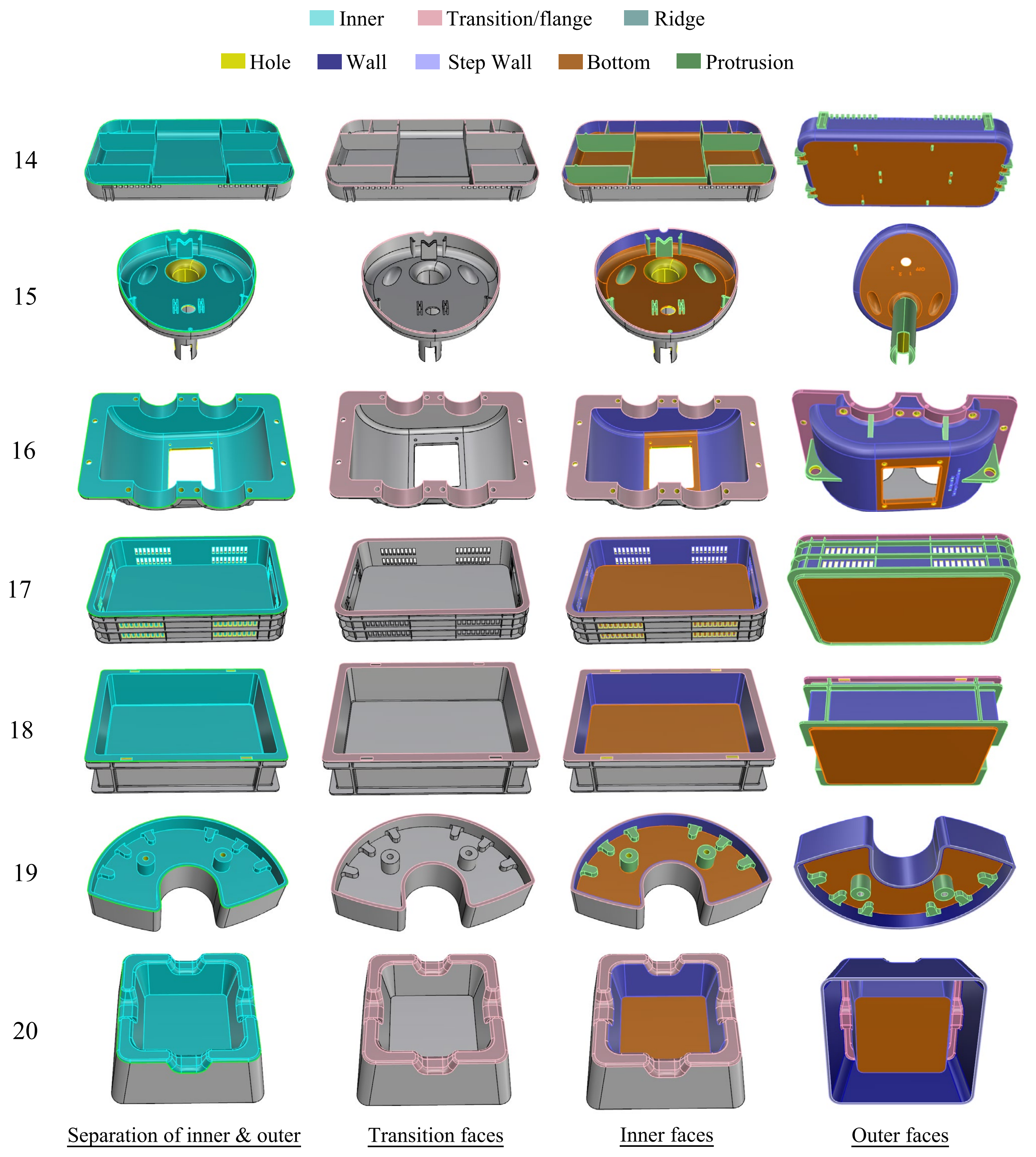


Fig. 16 Results of inner and outer faces recognition- complex structure outside

the accuracy of the proposed algorithm. For inner faces, only total number of faces are listed as the success rates for all cases are 100%. Outer faces listed include wall, flange, bottom and protrusion faces. The first and second values on each field represent total number of faces and number of faces not recognized correctly, respectively. Table 1 indicates that the summation of through holes, inner and outer faces is equal to total faces for all cases. This result is very important as it indicates that all faces on a model have been recognized as one of the face types. It would become easy to

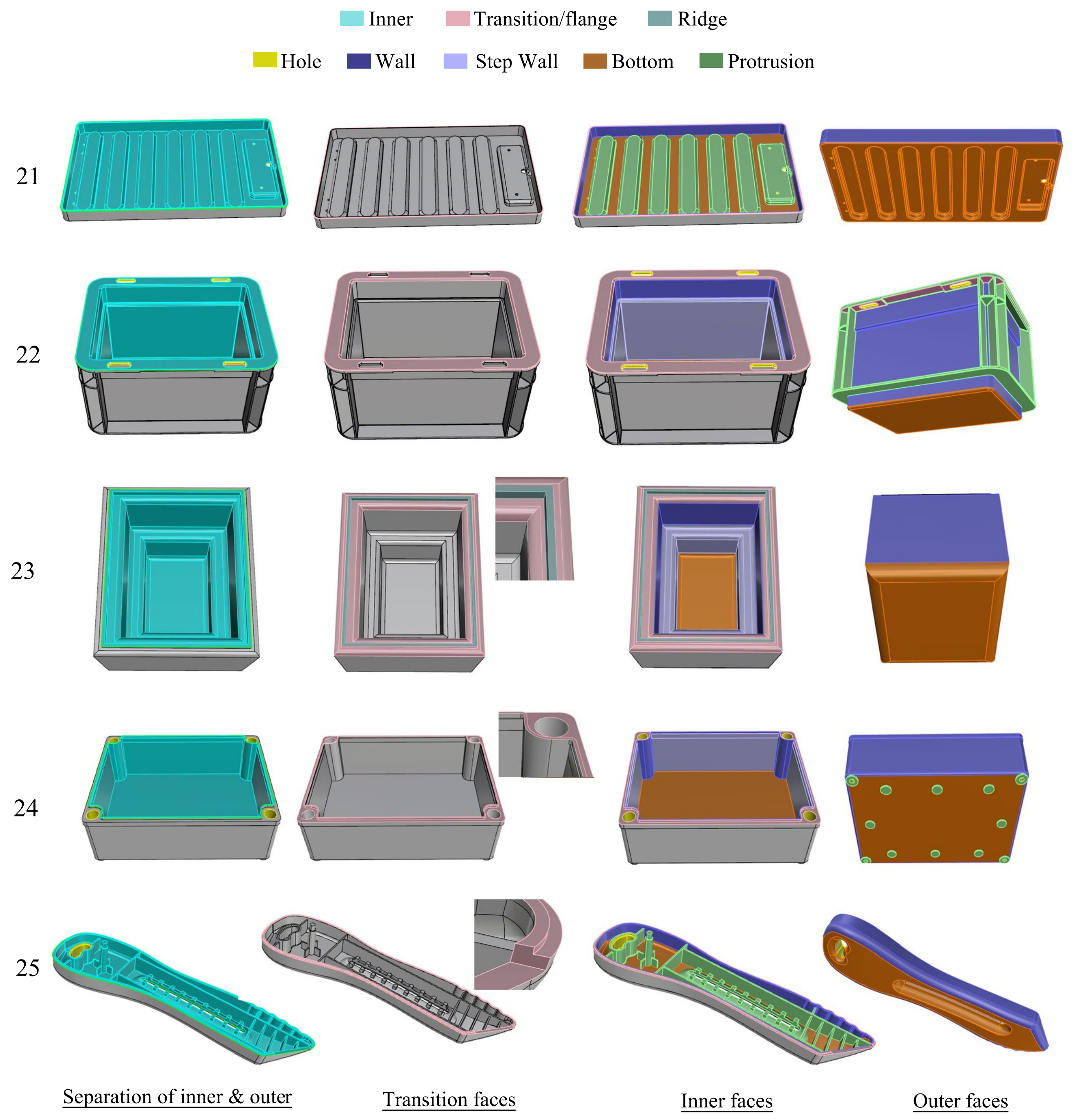


Fig. 17 Results of inner and outer faces recognition- different types of transition faces

look for the matching faces between the inside and outside of the thin shell on a model. Of 25 cases tested, the success rate for four of them, namely Cases 10, 14, 16 and 22, are not 100%, which is because the composition of the face types on outer faces is more complex and variable. Figure 18 depicts four examples of outer face types that are not recognized correctly. In Fig. 18a, f_i is a long and narrow surface lying across outer wall and bottom faces. The surface normal is changed across the entire surface. However, only one sampling point is taken for the projecting-line intersection check. It might be necessary to take more sampling points on a surface and modify the criteria to determine the status of the surface correctly. In Fig. 18b, f_i is a fillet between outer wall and bottom faces. As the fillet is also a surface, the problem is like that of Fig. 18a. In Fig. 18c, the blending faces between protrusion and flange faces are divided into

Table 1 Compositions of inner and outer faces and CPU time for 25 thin-shell parts

Case	Total faces	Through faces	Inner faces (Total/Fail)	Outer faces (Total/Fail)				Success rate (%)	CPU time (s)
				Wall	**Flange**	**Bottom**	**Protrusion**		
1	134	0	127/0	6/0	0/0	1/0	0/0	100	0.16
2	620	86	387/0	8/0	0/0	13/0	126/0	100	2.708
3	330	40	285/0	4/0	0/0	1/0	0/0	100	0.602
4	107	20	78/0	8/0	0/0	1/0	0/0	100	0.243
5	95	0	92/0	2/0	0/0	1/0	0/0	100	0.151
6	362	151	159/0	28/0	0/0	24/0	0/0	100	1.975
7	384	95	189/0	8/0	0/0	63/0	29/0	100	1.099
8	292	17	226/0	48/0	0/0	1/0	0/0	100	1.585
9	411	89	279/0	18/0	0/0	13/0	0/0	100	2.074
10	681	4	232/0	237/0	45/7	14/0	149/5	98.2	4.064
11	245	0	45/0	8/0	0/0	10/0	182/0	100	1.397
12	415	384	17/0	5/0	0/0	6/0	0/0	100	1
13	287	69	165/0	34/4	0/0	19/0	0/5	100	1.005
14	367	0	82/0	18/0	0/0	13/8	254/0	97.8	2.44
15	193	26	133/0	10/0	0/0	10/0	14/0	100	0.688
16	718	80	19/0	508/0	18/0	1/0	92/16	97.8	14.395
17	845	432	10/0	76/0	24/0	1/0	302/0	100	5.843
18	383	16	18/0	32/0	32/0	9/0	276/0	100	3.421
19	185	4	67/0	47/0	0/0	1/0	54/0	100	0.768
20	259	0	121/0	16/0	112/0	1/0	0/0	100	2.568
21	269	14	108/0	13/0	0/0	134/0	0/0	100	2.088
22	873	96	59/0	44/0	35/24	9/0	630/0	97.3	12.514
23	50	0	41/0	4/0	0/0	5/0	0/0	100	0.212
24	140	12	71/0	24/0	0/0	9/0	24/0	100	0.458
25	366	19	282/0	53/0	0/0	12/0	0/0	100	1.84

multiple pieces. The face composition is too complex to be recognized correctly. Similar situation occurs in Fig. 18d, where the face composition on the highlighted region is too complex, and hence the faces are not recognized correctly.

The CPU time required for the entire process for all cases is also listed in Table 1. The CPU time required is not fully proportional to total number of faces on the model. Sect. 4.4.1, initially separate outer wall and bottom faces, requires the maximum percentage of CPU time as this step has a complex procedure to check each outer face one by one. Therefore, Cases 16, 17 and 22 require more CPU time as the outer/total face of them are 613/718, 403/845 and 718/873, respectively. The simulations were performed on a personal computer with an Intel Core i7-9700 CPU 3.2 GHz and 16 GB of RAM.

6 Conclusion

An enhanced approach for inner and outer faces recognition of thin-shell parts was proposed in this study. A thin-shell part can be divided into a thin shell and protrusions that locate both on the inside and outside of the thin shell. The face types on the thin shell are recognized first. The complexity of the parts that can be dealt with includes: (1) five types of transition faces between inner and outer faces are detected. The composition of faces on the top of different thin-shell parts can be analyzed accurately; (2) protrusion faces that distribute continuously or sparsely both on the inside and outside of a part can be recognized; (3) fillets are also assigned as appropriate face types in accordance with the adjacency relationship. No simplification of fillets is necessary in this algorithm; and (4) holes that locate across multiple faces are allowed. It can enhance the applicability of the proposed algorithm for cases that involve holes across multiple faces. With the proposed method, all complex protrusion structures on the inside and outside of a part can be divided into groups of

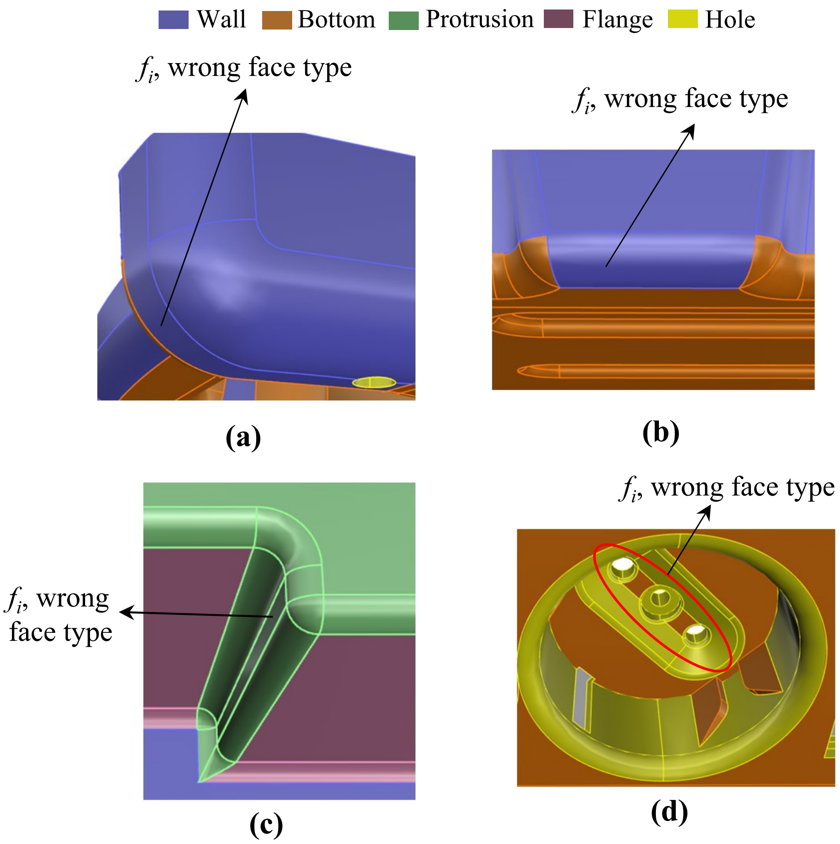


Fig. 18 Four examples of incorrect outer face types recognition, **a** example 1, **b** example 2, **c** example 3, and **d** example 4

protrusion face. A protrusion classification algorithm can later be used to recognize all types of protrusion [22].

The results of the face type recognition for the thin shell can further be used in volume decomposition. The primary task in volume decomposition is to decompose a model into a series of sweepable sub-volumes so that each of them can be meshed with better type of solid meshes. In [23], an approach by evaluating the matching pairs of inner and outer contours was proposed to decompose the thin shell of a thin-shell part into sweepable sub-volumes. It employs the face types on the inner and outer faces to help the evaluation of inner and outer matching contours. As mentioned previously, the composition of faces on a thin-shell part could vary significantly. The face type recognition method proposed in this study can analyze the composition of faces for complex thin-shell parts. It is now possible to expand the capability of the method in [23] to deal with more complex thin-shell parts.

Declarations

Conflict of interest The authors declare no relevant financial or conflict of interest to disclose.

References

1. Lu Y, Gadh R, Tautges TJ (2001) Feature based hex meshing methodology: feature recognition and volume decomposition. Comput Aided Des 33:221–232
2. Wu H, Gao S (2014) Automatic swept volume decomposition based on sweep directions extraction for hexahedral meshing. 23rd International Meshing Roundtable. Procedia Eng 82:136–148
3. Huang J, Tong Y, Weu H, Bao H (2011) Boundary aligned smooth 3D cross-frame field. ACM Trans Graph 30(6):1–8
4. Nieser M, Reitebuch U, Polthier K (2011) Cubecover- parameterization of 3D volumes. Computer Graph Forum 30(5):1397–1406
5. Li Y, Liu Y, Xu W, Wang W, Guo B (2012) All-hex meshing using singularity-restricted field. ACM Trans Graph 31(6):1–11
6. Howalski N, Ledoux F, Frey P (2016) Smoothness driven frame field generation for hexahedral meshing. Comput Aided Des 72:65–77
7. Hu K, Zhang YJ (2016) Centroidal Voronoi tessellation based polycube construction for adaptive all-hexahedral mesh generation. Comput Methods Appl Mech Eng 305:405–421
8. Hu K, Zhang YJ, Liao T (2017) Surface segmentation for polycube construction based on generalized centroidal Voronoi tessellation. Comput Methods Appl Mech Eng 316:280–296
9. Yu Y, Wei X, Li A, Liu JG, He J, Zhang YJ (2020) HexGen and Hex2Spline: polycube-based hexahedral mesh generation and spline modeling for isogeometric analysis applications in LS-DYNA. In: Springer INdAM Serie: Proceedings of INdAM Workshop "Geometric Challenges in Isogeometric Analysis".
10. Robinson TT, Armstrong CG, Fairey R (2011) Automated mixed dimensional modelling from 2D and 3D CAD models. Finite Elem Anal Des 47(2):151–165

11. Makem JE, Armstrong CG, Robinson TT (2014) Automatic decomposition and efficient semi-structured meshing of complex solids. Eng Comput 30:689–701
12. Nolan DC, Tierney CM, Armstrong CG, Robinson TT, Makem JE (2013) Automatic dimensional reduction and meshing of stiffened thin-wall structures. Eng Comput 30:689–701
13. Sun L, Tierney CM, Armstrong CG, Robinson TT (2016) Automatic decomposition of complex thin walled CAD models for hexahedral dominant meshing. Proc Eng 163:225–237
14. Sun L, Tierney CM, Armstrong CG, Robinson TT (2018) An enhanced approach to automatic decomposition of thin-walled components for hexahedral-dominant meshing. Eng Comput 34:431–447
15. Yu Y, Liu JG, Zhang YJ (2021) HexDom: polycube-based hexahedral dominant mesh generation. In: The Edited Volume of Mesh Generation and Adaptation: Cutting-Edge Techniques for the 60th Birthday of Oubay Hassan, SEMA-SIMAI Springer Series.
16. Woo Y (2003) Fast cell-based decomposition and applications to solid modeling. Comput Aided Des 35:969–977
17. Sundararajan V, Wright PK (2004) Volumetric feature recognition for machining components with freeform surfaces. Comput Aided Des 36:11–25
18. Lim T, Medellin H, Torres-Sanchez C, Corney JR, Ritchie JM, Davies JBC (2005) Edge-based identification of DP-features on free-form solids. IEEE T Pattern Anal Mach Intell 27(6):851–60
19. Sunil VB, Pande SS (2008) Automatic recognition of features from freeform surface CAD models. Comput Aided Des 40:502–517
20. Wang J, Wang Z, Zhu W, Ji Y (2010) Recognition of freeform surface machining features. J Comput Inf Sci Eng 10(4):041006
21. Lai JY, Wang MH, Song PP, Hsu CH, Tsai YC (2018) Automatic recognition and decomposition of rib features in thin-shell parts for mold flow analysis. Eng Comput 34:801–820
22. Lai JY, Song PP, Hsiao AS, Tsai YC, Hsu CH (2021) Recognition and classification of protrusion features on thin-wall parts for mold flow analysis. Eng Comput 37:833–854
23. Lai JY, Wu JW, Song PP, Chou TY, Tsai YC, Hsu CH (2022) Hybrid mesh generation for the thin shell of thin-shell plastic parts for mold flow analysis. Eng Comput 38:4895–4917
24. Joshi S, Chang TC (1988) Graph-based heuristics for recognition of machined features from a 3D solid model. Comput Aided Des 20(2):58–66
25. Corney J, Clark DER (1991) Method for finding holes and pockets that connect multiple faces in 2 1/2D objects. Comput Aided Des 23(10):658–668
26. Bruzzone E, Floriani LD (1991) Extracting adjacency relationships from a modular boundary model. Comput Aided Des 23(5):344–355
27. Waco DL, Kim YS (1994) Geometric reasoning for machining features using convex decomposition. Comput Aided Des 26(6):477–489
28. Sakurai H, Dave P (1996) Volume decomposition and feature recognition, Part II: curved objects. Comput Aided Des 28(6–7):519–537
29. Woo Y, Kim SH (2014) Protrusion recognition from solid model using orthogonal bounding factor. J Mech Sci Technol 28(5):1759–1764
30. Liu SS, Gadh R (1998) Basic logical bulk shapes (BLOBs) for finite element hexahedral mesh generation to support virtual prototyping. J Manu Sci Eng 120(4):728–735
31. Gadh R, Prinz FB (1995) A computationally efficient approach to feature abstraction in design-manufacturing integration. J Eng Ind 117:16–27
32. Sheen DP, Son TG, Myung DK, Lee SH, Lee K, Yeo TJ (2010) Transformation of a thin-walled solid model into a surface model via solid deflation. Comput Aided Des 42:720–730
33. Zhu H, Shao Y, Liu Y, Zhao J (2016) Automatic hierarchical midsurface abstraction of thin-walled model based on rib decomposition. Adv Eng Softw 97:60–71
34. White DR, Saigal S, Owen SJ (2004) CCSweep: automatic decomposition of multi-sweep volumes. Eng Comput 20:222–236
35. Cai S, Tautges TJ (2015) One-to-one sweeping based on harmonic S-T mappings of facet meshes and their cages. Eng Comput 31:439–452
36. Wu H, Gao S (2014) Automatic swept volume decomposition based on sweep directions extraction for hexahedral meshing. 23rd International Meshing Roundtable. Procedia Eng 82:136–148
37. Luo XJ, Shephard MS, Yin LZ, O'Bara RM, Nastasi R, Beall MW (2010) Construction of near optimal meshes for 3D curved domains with thin sections and singularities for p-version method. Eng Comput 26:215–229
38. Liu L, Zhang Y, Liu Y, Wang W (2015) Feature-preserving T-mesh construction using skeleton-based polycubes. Comput Aided Des 58:162–172
39. Lai JY, Wang MH, You ZW, Chiu YK, Hsu CH, Tsai YC, Huang CY (2016) Recognition of virtual loops on 3D CAD models based on the B-rep model. Eng Comput 32:592–606